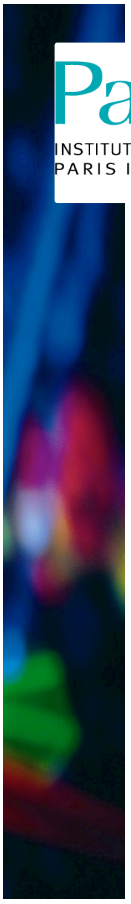


ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY

TELECOM ParisTech

2011-TELP-00XX

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

Télécom ParisTech

Spécialité " Signal et Images "

présentée et soutenue publiquement par

Matthieu MOINARD

le 1 juillet 2011

Codage vidéo hybride basé contenu par analyse/synthèse de données

Directeur de thèse : **Pierre DUHAMEL**
Co-encadrement de la thèse : **Isabelle AMONOU, Patrice BRAULT**

Jury

M. Rémy PROST, Professeur des Universités, Creatis, Insa Lyon	Rapporteur
M. Joseph RONSIN, Professeur des Universités, IETR, Insa Rennes	Rapporteur
M. Claude LABIT, Directeur de Recherche, INRIA Rennes	Examinateur
Mme Béatrice PESQUET-POPESCU, Professeur des Universités, Télécom ParisTech	Examinateur

T H È S E

Télécom ParisTech
Grande école de l'Institut Télécom – membre fondateur de ParisTech
46, rue Barrault – 75634 Paris Cedex 13 – Tél. + 33 (0)1 45 81 77 77 – www.telecom-paristech.fr

Avant Propos

Ce travail a été effectué dans le cadre d'un partenariat entre le Laboratoire des Signaux et Système (UMR 8506) et la division Recherche & Développement d'Orange Labs. De ce fait, ces trois années de recherche m'ont permis de côtoyer deux univers différents, qui se rejoignent néanmoins sur la richesse des rencontres que l'on y fait. A ce titre, je remercie les nombreuses personnes qui, de près ou de loin, ont participé à cette aventure.

Je tiens à exprimer toute ma reconnaissance à mon directeur de thèse, Pierre Duhamel, pour ses conseils avisés. Sa perspicacité et sa clairvoyance auront été une aide précieuse pour mener à bien ce travail de recherche.

Je remercie également Patrice Brault pour l'intérêt qu'il a porté à cette étude. Son investissement et sa ténacité auront permis, entre autres, de défricher des terrains scientifiques des plus hostiles. Son entrain communicatif a été d'une grande valeur. Travailler en sa compagnie n'en a été que plus agréable.

Tous ceux qui ont partagé mon quotidien pendant ces trois années savent à quel point je remercie Isabelle Amonou pour sa disponibilité et sa confiance. Les innombrables heures à partager son savoir et son expérience me seront à jamais précieuses. Je lui suis reconnaissant de m'avoir initié au monde de la recherche en entreprise et de m'avoir transmis le virus.

Je tiens à remercier chaleureusement Rémy Prost et Joseph Ronsin d'avoir assumé la charge de rapporteur. Je veux de plus leur manifester ma très sincère reconnaissance quant au sérieux et à l'esprit critique dont ils ont fait preuve dans cet exercice. J'exprime également toute ma gratitude envers Claude Labit et Béatrice Pesquet-Popescu pour avoir accepter d'évaluer mon travail.

A toutes les autres personnes qui ont collaboré et contribué à ces travaux, d'une façon ou d'une autre, je leur témoigne toute mon admiration. Je dois beaucoup à leur investissement.

Je tire mon chapeau à tous les membres, anciens et présents, de l'équipe Codage Vidéo Avancé d'Orange Labs qui m'ont accueilli à bras ouverts. Merci de m'avoir immergé dans le vaste monde du codage vidéo et de m'en avoir expliqué tous les rouages. La traditionnelle pause café matinale à philosopher sur tout et rien me manque déjà !

A titre personnel, je souhaite remercier ma famille, et notamment mes parents pour leur dévouement sans faille depuis toutes ces années.

Merci à tous mes proches et amis, où qu'ils se trouvent, pour leurs soutiens et bien d'autres choses.

Enfin, mes derniers remerciements seront pour Bleuenn, qui m'a encouragé depuis le début de cette aventure et qui a eu un rôle ô combien important, bien qu'elle vous affirmera le contraire !

A toute ces personnes et à bien d'autres, un grand merci.

Table des matières

Table des figures

Liste des Acronymes

ATM	*Averaged Template Matching*
AVC	*Advanced Video Coding*
BAB	*Binary Alpha Block*
bpp	Bit par pixel
CABAC	*Context-based Adaptive Binary Arithmetic Coding* Codage arithmétique avec utilisation d'un contexte adaptatif
CAVLC	*Context-Adaptive Variable-Length Coding*
CIF	*Common Intermediate Format* Résolution de séquence vidéo 352×288 pixels
CM	Compensation de Mouvement
dB	Décibel
DCT	*Discrete Cosine Transform*
DVB	*Digital Video Broadcasting*
EDP	Equation aux Dérivées Partielles
EM	Estimation de Mouvement
FRExt	*Fidelity Range Extensions*
GOF ou GOP	*Group Of Frames*, *Group Of Pictures* Fait référence à une séquence d'images traitée d'un seul bloc
H.26x	H.261, H.262, H.263, H.26L sont des normes de compression vidéo numérique définies par ITU. H.264 est issue d'un travail conjoint entre ITU et ISO.
HEVC	*High Efficiency Video Coding* Norme de codage vidéo en cours d'élaboration succédant à H.264
INTER (P,B)	Mode de codage par utilisation de la prédiction temporelle simple (P) ou bidirectionnelle (B).
INTRA (I)	Mode de codage par prédiction spatiale exclusivement.
ISO	*International Organization for Standardization*

ISO	*International Standard Organization*
ITU	*International Telecommunications Union*
JPEG	*Joint Picture Expert Group* JPEG et JPEG2000 sont des normes de compression d'images fixes.
kb/s	kilo-bits par secondes
MPEG	*Motion Picture Expert Group*
MPM	*Most Probable Mode*
MSE	*Mean Square Error*
MVD	*Motion Vector Difference*
NAL	*Network Abstaction Layer*
PSNR	*Peak Signal to Noise Ratio* Mesure objective de qualité d'image et de vidéo
QCIF	*Quart de CIF* Résolution de séquence vidéo 176×144 pixels
RD-opt	*Rate-Distortion Optimization*
RD-opt	fonction d'optimisation débit-distorsion
SA-DCT	*Shape-Adaptive DCT*
SAD	*Sum of Absolute Differences*
SMPM	*Second Most Probable Mode*
SMPTE	*Society of Motion Picture and Television Engineers*
SSD	*Sum of Squared Differences*
STMP	*Set of Template Matching Predictors*
Texel	***tex**ture **el**ement* Motif élémentaire pour les textures déterministes
TM	*template matching*
TV	*Total Variation*
VOP	*Video Object Plane*
VQA	*Video Quality Assessor*

Introduction Générale

Les travaux de cette thèse s'inscrivent à la fois dans le contexte du **traitement d'image** et dans celui du **codage appliqué à la vidéo numérique**.

Le **traitement d'image numérique** sert un très grand nombre d'applications concrètes dans différents domaines. Nous pouvons par exemple citer la reconnaissance automatique d'écriture, de visage ou de mouvement, l'imagerie médicale, la détection d'objets dans les images satellites, la visio-conférence, le rendu 3D cinématographique, etc. En ce qui nous concerne, deux sous domaines du traitement d'image sont particulièrement visés : l'**analyse/synthèse** d'images naturelles et la **restauration d'image**.

L'**analyse/synthèse de texture** consiste à caractériser une texture source à partir d'une quantité restreinte de données, grâce auxquelles le procédé de synthèse créé une texture visuellement "similaire" à la texture source.

La **restauration d'image** est un procédé destiné à améliorer une image. Elle regroupe plusieurs champs d'applications tels que la suppression de bruit et/ou de flou, la correction d'erreurs dûes à un canal de transmission bruité, la suppression d'objets indésirables dans une scène, l'effacement d'artefacts liés à la méthode de compression utilisée, etc.

De nombreuses recherches ont porté sur ces deux domaines et plus particulièrement au cours de ces dernières années. L'amélioration des procédés, couplée à l'augmentation de la puissance de calcul disponible, a contribué à l'émergence de ces outils de traitements automatisés. Nous proposons d'étudier quelques-unes de ces méthodes de traitement d'image dans le but d'élaborer plusieurs modèles de prédiction d'image naturelle qui, une fois intégrés dans un schéma de codage de vidéo, permettent d'en améliorer le rendement en compression.

Le **codage** de séquence vidéo se justifie par l'économie des ressources de stockage et des canaux de télécommunication qu'il entraîne. Nous nous intéressons notamment au cas du codage avec perte d'information, communément employé dans les applications de télécommunications. Les performances se mesurent en terme de compromis débit/distorsion, c'est-à-dire que pour un volume d'information donné nous cherchons à maximiser la qualité de la séquence décodée en mesurant sa qualité par rapport à la vidéo d'origine. A l'heure actuelle, les algorithmes de codages vidéo existants montrent leurs limites dans leur inefficacité à compresser certains types de contenu d'image, notamment en ce qui concerne les textures complexes.

La démarche de notre étude consiste à nous inspirer de certaines méthodes issues du traitement d'image pour les transposer au cadre formel du codage vidéo. Nous proposons en particulier deux nouvelles méthodes prédictives directement exploitables au sein d'un codeur/décodeur vidéo. L'intérêt de ce travail est l'amélioration des performances en compression des algorithmes de codage vidéo

actuels. C'est pourquoi ce document s'adresse en particulier à la communauté du codage vidéo ; nous essayons ici d'apporter de nouveaux outils de codage vidéo en nous inspirant des domaines de recherche connexes du traitement d'image.

Nous proposons de poursuivre l'introduction de ce rapport dans le chapitre suivant dédié à l'environnement et aux enjeux actuels du codage vidéo, puis de présenter les contributions de ce projet de thèse. La partie suivante est dédiée à la description technique des outils de codage vidéo. Cette étude apporte les connaissances nécessaires à la compréhension des travaux de recherche détaillés dans la suite de ce manuscrit.

Première partie

Préliminaires

CHAPITRE 1

Contexte général et motivations

Sommaire

1.1 Introduction

La vidéo numérique a su s'imposer au détriment de l'analogique grâce à une qualité et à un niveau de service bien supérieur à tous les niveaux de la chaîne de traitement d'un flux vidéo : acquisition, stockage, transmission et restitution. Historiquement, la vidéo numérique s'est surtout démocratisée auprès du grand public grâce à l'arrivée simultanée des premières offres commerciales de télévision numérique par satellite et par câble, et du DVD [1] vidéo lancé officiellement en 1995. Ce support permet d'exploiter une image de meilleure qualité par rapport à ses homologues analogiques, tout en ajoutant de nouvelles fonctionnalités telles que l'accès aléatoire, les bonus, les pistes audio et les sous-titrages en plusieurs langues, *etc.* Le succès du DVD vidéo est en partie attribué au format de codage vidéo utilisé qui a permis d'atteindre un facteur de compression suffisant pour promouvoir son utilisation. Nous aurons l'occasion de revenir sur ce point dans ce chapitre. Le successeur reconnu du DVD, le disque Blu-ray, dont les premières spécifications datent de 2002, permet de stocker toujours plus de données et donc d'offrir une meilleure qualité visuelle. Il s'agit dans ce cas de vidéo haute-définition (HD), et même récemment de vidéo stéréoscopique simulant une perception de relief sur les appareils compatibles. En parallèle, le déploiement de la Télévision Numérique Terrestre (TNT) en France et dans beaucoup de pays étrangers, qui permet la diffusion numérique de la télévision *via* le réseau hertzien, a définitivement achevé l'ère de la vidéo analogique.

Une nouvelle étape a été franchie au cours de ces dernières années avec l'essor de la dématérialisation des contenus audiovisuels au détriment des supports physiques, ce qui se traduit par la diffusion de

1. *Digital Versatile Disc*

ces contenus sur les réseaux de télécommunication, en particulier Internet. Les avantages de la dématérialisation sont multiples. Elle permet, pour le consommateur, d'accéder instantanément à un catalogue abondant et actuel de contenus, mis à jour de façon transparente. Pour l'éditeur, cette procédure est plus flexible et économique, puisqu'elle ne nécessite pas de production physique des supports ni de coût d'acheminement, ce qui est, de plus, un facteur environnemental non négligeable.

Face à cette nouvelle tendance, les opérateurs de télécommunication deviennent les garants du déploiement et de l'exploitation des services émergents liés à la dématérialisation des supports physiques. C'est pourquoi ils sont au coeur des enjeux actuels et à venir.

La section suivante définit la problématique liée à l'augmentation du trafic des télécommunications et illustre les changements de mode de consommation responsables de cette envolée. Ensuite, nous nous intéresserons aux enjeux de cette transformation, pour les opérateurs de télécommunication, et à la nécessité d'explorer de nouvelles techniques de codage. Suite à cela, nous énumérerons les contributions réalisées dans le cadre de ce projet de thèse et présenterons l'organisation de ce document.

1.2 Problématique

Cette partie analyse la problématique industrielle en évoquant la place croissante de la vidéo numérique dans notre société. Elle a pour objectif d'introduire les enjeux de cette thèse. Nous essaierons de décrypter les nouvelles tendances d'usage et de consommation vidéo, qui conditionneront les défis de la prochaine décennie.

Nous assistons depuis ces dernières années à l'envolée du trafic Internet. Les prévisions pour ces prochaines années confirment cette croissance avec un TCAC [2] du trafic de 36% jusqu'en 2014, comme illustré sur la figure 1.1. Outre cette tendance, nous constatons que la proportion du volume des données transmises dédiées à la vidéo augmente considérablement entre 2009 et 2014 par rapport aux autres types de données transitant sur le réseau Internet. En cumulant tous les usages basés sur la transmission de vidéo, à savoir la vidéo sur Internet, la télévision par Internet, les échanges P2P [3] de vidéo et les appels vidéo, la transmission de vidéo sur le réseau Internet dépassera 91% du trafic grand public mondial d'ici 2014 (source : [11]).

Cette explosion de la diffusion de contenu vidéo s'explique par un changement des usages classiques de consommation, rendu possible grâce au déploiement de nouvelles technologies. Ainsi, l'utilisateur devient plus exigeant quant au choix et à l'accès du contenu. La demande des consommateurs s'oriente vers les préceptes de "*ce que je veux*", "*où je veux*", "*quand je veux*" et "*comme je veux*".

Le "*ce que je veux*" montre un changement en profondeur dans la façon d'appréhender ce type de média : le consommateur sort de sa condition passive de téléspectateur où son choix se limite à la sélection du canaux de diffusion *live*. Nous passons alors de média de masse à des systèmes hyperspécialisés où chaque utilisateur choisit le contenu qu'il désire, sans aucune contrainte de temps. Les éditeurs et diffuseurs de média audiovisuel doivent s'adapter à ces nouvelles exigences en diversifiant les contenus et en facilitant leurs accès. Pour cela, les canaux de diffusion de la télévision se multiplient : IPTV par ADSL ou fibre optique, télévision par satellite ou par le réseau hertzien (TNT), et des services innovants qui émergent : *VOD* (*Video On Demand*), télévision de rattrapage, *timeshifting* (ou contrôle du direct).

2. Taux de Croissance Annuel Composé
3. Peer-to-peer

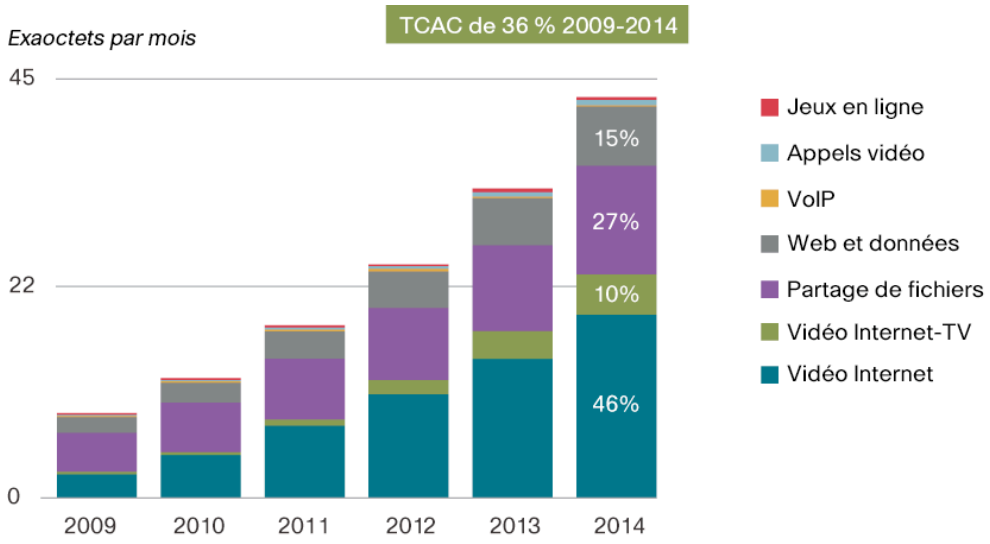


FIGURE 1.1 – Prévisions du trafic Internet grand public mondial (source : [11]).

La prolifération des contenus s'explique aussi par le développement très rapide d'un usage récent associé aux nouvelles technologies, qui change du rapport traditionnel et figé de producteur/consommateur : le Web 2.0, Web participatif ou encore Web social. Dans ce nouveau système, les deux entités fusionnent ; nous constatons alors que le consommateur devient à son tour créateur de contenu numérique. Dans le cas de la vidéo, nous citerons notamment le succès indéniable de la plateforme communautaire YouTube d'hébergement de contenu vidéo. A titre de comparaison, YouTube compte deux fois plus d'audience que le *prime-time* des trois plus gros diffuseurs de télévisions aux Etats-Unis réunis (*ABC*, *CBS* et *NBC*) et a une portée internationale (données issues de http ://www.youtube.com/t/fact_sheet).

Cette métamorphose du système a été rendue possible grâce à l'avènement des appareils de captation audiovisuels toujours plus performants. En 2010, la possibilité de filmer avec son téléphone n'est plus réservée aux modèles haut de gamme, et beaucoup d'appareils photos grand public sont capables d'enregistrer une vidéo au format HD.

Le "*où je veux*" et "*quand je veux*" sont liés à la notion d'hyperconnectivité. L'utilisateur veut être de plus en plus connecté. Pour cela, il exploite de plus en plus d'appareils électroniques différents, souvent simultanément, pour accéder aux média numériques : téléviseur, téléphone portable, ordinateur, tablette... avec des technologies de réception propres. L'impact de l'hyperconnectivité se fait ressentir à travers l'augmentation du trafic IP dans le monde, d'autant plus que beaucoup de services émergents ont pour support la vidéo. Cette tendance a un fort impact sur la charge réseau des opérateurs, avec par exemple l'appel par visioconférence ou les réseaux sociaux. Les fournisseurs d'accès à l'Internet doivent s'adapter à cette tendance en densifiant leurs infrastructures réseaux et en améliorant la qualité de service du trafic de données, notamment mobile. En effet, on estime que 66% du trafic de données mobiles mondial sera de la vidéo d'ici 2014, ce qui représente un taux de croissance annuel composé (TCAC) de 131% entre 2009 et 2014 (source [11]), dû en partie à un effet de "glissement" des dispositifs portables, notamment les ordinateurs, vers le réseau mobile.

Enfin, le "*comme je veux*" fait référence au caractère multi-aspect d'un même contenu vidéo. Les progrès technologiques et la baisse des coûts de production ont permis d'offrir la possibilité au grand public d'acquérir des dispositifs d'affichage HD (haute-définition) ou 3D (permettant la restitution

d'images stéréoscopiques). Par exemple, la surface totale de tous les écrans numériques du monde sera 1, 7 fois supérieure en 2014 par rapport à fin 2009 (source [11]), augmentant la résolution moyenne des écrans selon la même proportion. Pour exploiter ces nouveaux écrans, les diffuseurs de contenu audiovisuel doivent être en mesure de proposer un flux vidéo adapté au dispositif de l'utilisateur. Ces formats de vidéo avancée (HD et 3D) sont beaucoup plus gourmands en terme de bande-passante, ce qui augmente considérablement le trafic des vidéos sur les réseaux de télécommunication.

Pour l'avenir, un grand nombre des acteurs concernés prédisent une accélération du phénomène de dématérialisation qui s'étendra à d'autres types d'applications, notamment à l'univers vidéoludique à travers le *cloud gaming*. Assurément, l'expansion de ce nouveau type de service va à terme accroître le trafic de contenu vidéo sur les réseaux de télécommunications, déjà impactés par la généralisation de la vidéo HD et stéréoscopique ("3D").

1.3 Enjeux et retombées

Face à la montée en puissance d'Internet et des nouveaux usages nomades, les prochains défis de l'ère numérique concerneront en particulier les opérateurs de télécommunication, quelque soit le support permettant de véhiculer l'information (câble, fibre optique, onde radioélectrique). Les opérateurs de télécommunication doivent répondre à ces nouveaux besoins en agissant sur deux facteurs : augmenter la capacité du canal de transmission et réduire le volume de données transitant sur les réseaux en assurant une qualité de service équivalente. Cette deuxième possibilité repose sur l'amélioration des algorithmes de codage dans le but d'obtenir un facteur de compression plus important. La vidéo numérique atteindra 91% du volume de données transitant sur l'Internet aux environs de 2014. Elle représentera, parmi tous les types de contenu, l'axe de recherche principal qui permettra à terme de décongestionner les réseaux. Pour cela, les progrès techniques réalisés en terme d'efficacité des appareils électroniques, permettent d'envisager des méthodes de codage vidéo toujours plus complexes et performantes.

Pour un opérateur de télécommunication, à l'instar d'Orange, les retombées de tels projets de recherche sont multiples. La réduction de bande passante obtenue grâce au perfectionnement des outils d'encodage minimise l'investissement réalisé dans les infrastructures de réseaux pour en augmenter les capacités. De ce point de vue, les retombées sont financières et, dans une moindre mesure, environnementales. De plus, la recherche dans ce domaine apporte une expertise et un patrimoine intellectuel valorisé par des brevets.

C'est dans ce contexte que s'inscrivent les travaux de cette thèse.

1.4 Contributions

Les recherches effectuées dans le cadre de ce projet de thèse visent essentiellement à améliorer les performances en compression des méthodes de codage vidéo conventionnelles, et en particulier celles de la norme de codage H.264/AVC dont la dernière version date de 2005. Pour cela, nous nous sommes inspirés de certains procédés issus du traitement d'image que nous avons incorporés au sein des codeurs et décodeurs vidéo. Deux axes de recherche distincts ont été explorés :

– Le premier est fondé sur les méthodes d'analyse et de synthèse de texture telles qu'elles sont définies pour les images naturelles. Ce type d'outil est utilisé dans un contexte de codage vidéo pour prédire une portion de texture en fonction de la connaissance *a priori* de cette dernière. Ceci permet d'accroître le facteur de compression par rapport aux méthodes classiques de prédiction spatiale, peu efficaces dans le cas des textures. D'autres travaux antérieurs ont été menés sur ce sujet et regroupés sous la notion de *template matching*. Nous avons cherché à améliorer ces méthodes en prenant en compte les spécificités du codage vidéo de type H.264/AVC, et en particulier la fonction débit/distorsion basée sur une mesure objective des dégradations qui est au premier abord peu compatible avec la notion de synthèse de texture.

– Le deuxième s'inspire des méthodes de régularisation d'image, et notamment celles basées sur la minimisation de la variation totale. Ces méthodes ont été élaborées dans le but d'améliorer la qualité d'une image en fonction de la connaissance *a priori* des dégradations qu'elle a subies. Nous nous sommes basés sur ces réflexions pour proposer une nouvelle méthode de prédiction des coefficients transformés obtenus à partir d'une image naturelle.

Ces travaux se sont appuyés sur des concepts propres au traitement d'image fixe, c'est pourquoi nos recherches sont destinées avant tout à la modélisation spatiale des images dans le but d'aboutir à des outils de prédiction efficaces. Cependant, nous nous sommes efforcés de faire le parallèle avec les modèles d'estimation de mouvement utilisés dans le codage vidéo pour chacun des deux axes étudiés. L'objectif est d'amener de nouvelles pistes de réflexion sur une extrapolation de nos recherches en vu de les adapter aux aspects temporels du codage vidéo.

L'objectif principal des travaux porte sur les performances en compression des méthodes de codage vidéo. Nous débattons également de la complexité calculatoire des nouveaux outils que nous proposons pour qu'ils soient acceptables, c'est-à-dire que le ratio compression/complexité soit optimal.

1.5 Organisation du document

Le chapitre suivant est dédié à un état de l'art du codage vidéo numérique. Après l'introduction générale des concepts de codage, le chapitre présente la description technique de la norme H.264/AVC nécessaire à la compréhension des différentes implémentations réalisées dans le cadre de cette thèse et exposées postérieurement. Ensuite, le rapport est scindé en deux parties, correspondant aux deux axes de recherche investigués. La première correspond aux méthodes de synthèse de texture appliquées au codage vidéo, elle est composée de deux chapitres présentés comme tels :

– Le **chapitre 3** définit la notion de texture d'une image et dresse un panorama non exhaustif des méthodes reconnues dans le domaine de la synthèse de texture. Ensuite sont exposées les méthodes de restauration d'image inspirées des algorithmes de synthèse de texture. Enfin, un état de l'art porte sur l'adaptation de ces procédés aux contraintes du codage vidéo numérique.

– Le **chapitre 4** présente notre contribution aux algorithmes de codage par *template matching* présentés dans le chapitre 3. Après la présentation détaillée de la méthode et de son implémentation au sein d'un codeur de type H.264/AVC, les résultats expérimentaux montrent l'intérêt de notre approche par rapport à l'état de l'art.

La deuxième partie est basée sur l'utilisation de méthodes de restauration et d'outils de régularisation d'image pour améliorer l'efficacité en compression des codeurs d'image fixe et de vidéo. En particulier :

– Le **chapitre 5** présente un état de l'art sur les méthodes de régularisation d'image basée sur la minimisation de la variation totale. Notamment, différentes applications possibles grâce à ce procédé sont illustrées tout en soulignant les changements apportés au modèle théorique. Nous nous intéressons notamment aux nouvelles approches de codage présentées dans la littérature qui s'inspirent de ce type de méthode de régularisation.

– Le **chapitre 6** est dédié à la présentation des travaux de recherche sur l'élaboration d'un modèle innovant de restauration de coefficients dégradés basé sur la minimisation de la variation totale. Ce modèle est ensuite décliné afin d'être utilisé comme un outil de prédiction dans un codeur d'image fixe (type JPEG) et de vidéo (type H.264/AVC). Les expérimentations sont présentées afin de juger de l'intérêt de la méthode.

CHAPITRE 2

Codage vidéo : environnement et état de l'art

Sommaire

2.1 Introduction

Les problèmes liés au codage vidéo constituent un thème de recherche qui, bien qu'exploré depuis de nombreuses années, demeure très actuel. La nécessité de recourir au codage vidéo s'explique par le volume très élevé d'information essentiel au stockage et à la transmission de ce média. Pour illustrer ces propos, prenons l'exemple d'une vidéo numérique haute définition, le standard de plus en plus actuel, composée de 1080 lignes et 1920 colonnes de pixels. Pour une vidéo "brute" (sans codage), à chaque pixel correspondent trois couleurs primaires, respectivement représentées sur 1 octet. Une seule image "pèse" donc $1920 \times 1080 \times 3$ octets, soit environ 48 Mb [1]. A raison de 25 images par seconde pour la télévision, nous atteignons le débit d'environ 1200 Mb/s, ce qui est inconcevable en considérant les moyens de télécommunication actuels. A titre de comparaison, se priver de codage vidéo reviendrait à stocker moins de 3 minutes sur un support Blu-ray de 25 Go [2], contre $2h30$ grâce à l'utilisation du codage vidéo. Le débit est alors de 23 Mb/s, soit un ratio de plus de 50 fois inférieur. Pour atteindre ces performances, la compression est dite *avec perte* ou *destructive*, c'est-à-dire que le signal décodé n'est pas exactement similaire à l'original, mais les distorsions sont la plupart du temps invisibles à l'oeil nu. A titre de comparaison, les méthodes de compression *sans perte* obtiennent un ratio en compression compris entre 2 et 3.

1. Mégabit
2. Gigaoctet

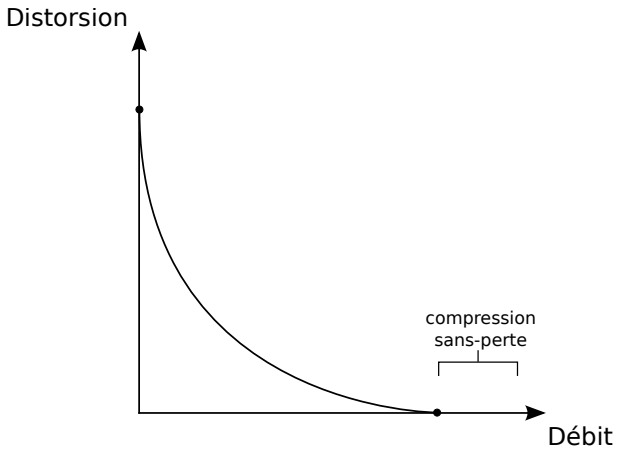


FIGURE 2.1 – Fonction débit-distorsion appliquée au codage vidéo.

L'efficacité d'un schéma de codage se mesure à l'aide d'un critère débit vs. distorsion : pour un volume de données faible nous souhaitons une qualité de reconstruction la plus élevée. La fonction débit-distorsion (Fig. 2.1) est alors définie pour exprimer l'un en fonction de l'autre : elle représente la distorsion minimale pour un débit donné et inversement. Le débit est mesurable par le nombre de bits nécessaires pour encoder la séquence. En revanche, il convient d'évaluer la distorsion selon un critère objectif, c'est-à-dire une comparaison pixel-à-pixel entre la séquence originale et décodée, ou subjectif en considérant des modèles psychovisuels plus élaborés s'appuyant sur l'étude du système visuel humain (voir [20] sur ce sujet). Ce dernier étant complexe et lourd à mettre en oeuvre, une métrique simple, le PSNR [3], sera préférée pour quantifier la distorsion introduite dans la séquence décodée. Le PSNR exprimé en dB est défini par :

$$PSNR = 10 \cdot \log_{10} \left(\frac{(MAX_I)^2}{EQM} \right) \tag{2.1}$$

où MAX_I caractérise la valeur maximale d'un pixel de l'image I. Typiquement, un pixel étant représenté sur 8 bits pour une image monochrome, nous avons $MAX_I = 2^8 - 1$, soit 255. L'erreur quadratique moyenne EQM est calculée entre deux images I_o et I_d de tailles $M \times N$ pixels selon :

$$EQM = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left(I_o(i,j) - I_d(i,j) \right)^2 \tag{2.2}$$

Grâce à la mesure de PSNR, nous disposons d'un outil simple et relativement efficace pour mesurer et comparer l'efficacité des différents outils de compression sur lesquels nous allons travailler. Quant aux modèles psychovisuels, basés sur l'étude du système visuel humain, la complexité de leurs mises en oeuvre au regard de l'amélioration des performances qu'ils confèrent rend leurs utilisations peu exploitées dans les schémas de codage actuels.

3. *Peak Signal to Noise Ratio*

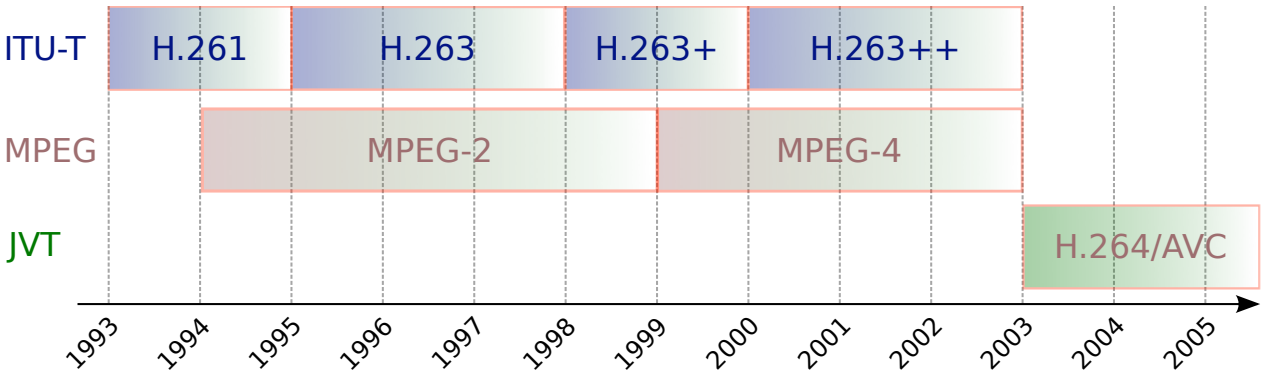


FIGURE 2.2 – Historique des principales normes de codage vidéo selon l'organisme de standardisation.

2.2 Les standards de codage vidéo numérique

2.2.1 Historique

La nécessité de définir des normes de compression vidéo s'est imposée lors de l'avènement des supports de stockage amovibles et l'accès aux télécommunications. Il était alors indispensable de décrire des méthodes universelles de codage/décodage pour permettre aux utilisateurs de visualiser une vidéo numérique indépendamment du matériel utilisé. Historiquement, deux grands organismes de standardisation de codage vidéo cohabitent : l'ITU-T (*International Telecommunication Union*) et le comité MPEG (*Moving Picture Experts Group*) créé en 1988 et dépendant de l'ISO/IEC (*International Organization for Standardization/International Electrotechnical Commission*). En 2001, la volonté de conjuguer leurs efforts donne naissance au groupe de travail JVT (*Joint Video Team*) dédié à la création et au développement de spécifications de nouveaux procédés de codage vidéo. L'historique des différentes familles de normes H.26x et MPEG-x établies par l'ITU-T, MPEG ou JVT est illustré sur la figure 2.2.

La norme H.261 [2] a été conçue pour les applications de visiophonie destinées au réseau RNIS (*Réseau Numérique à Intégration de Services*) à des débits multiples de 64 kbit/s et pour des résolutions d'image QCIF et CIF. Cette norme a été définie en utilisant les principaux procédés de compression encore largement employés à ce jour : estimation et compensation de mouvement, calcul et transformation DCT du résidu de prédiction, quantification et codage entropique. Le standard H.263 [3] visait à augmenter les performances en compression à très bas débit et a été successivement amélioré dans H.263+ [6] et H.263++ [7] pour prendre en compte de nouveaux profils d'utilisation tout en diminuant les débits d'encodage.

La norme MPEG-2 [4] a été notamment définie pour les applications liées à la télévision numérique grand public. Bien que datant de 1995, cette norme est encore exploitée à grande échelle puisqu'elle a été adoptée par le consortium DVB (*Digital Video Broadcasting*) pour les services de TV numérique par voie hertzienne terrestre et satellite. Elle est également employée comme format de codage du DVD. La norme MPEG-4 [1] définit un standard de compression générique prenant en charge la manipulation d'objets variés dans une scène permettant notamment une représentation sémantique d'une séquence. Elle englobe de nombreux procédés de codage regroupés en différentes parties dont chacune d'elle est dédiée à un type d'application particulier. Ainsi, une multitude d'applications multimédia peut être adressée, par exemple le téléchargement et le *streaming* via Internet, la visioconférence ou encore la réalité virtuelle distribuée.

Enfin, le rapprochement entre l'ITU-T et MPEG a abouti à la définition du standard H.264/AVC (AVC pour *Advanced Video Coding*) [8], aussi connu sous le nom de MPEG-4 Part 10, dont les premières spécifications furent publiées en 2003. La première extension, connue sous le nom de H.264/AVC FREext (*Fidelity Range Extensions*) [133] fut finalisée en 2004 et apporta de nouveaux outils d'encodage plus performants. De façon générale, la norme H.264/AVC ne propose pas de bouleversement profond des techniques de codage habituelles des précédentes normes, mais repose sur une multitude d'améliorations des procédés existants, ce qui permet au final d'obtenir un facteur de compression plus de deux fois supérieur à MPEG-2 et MPEG-4 (partie 2), avec un accroissement de la complexité du décodeur de l'ordre de quatre fois celui de MPEG-2. Les performances en compression couplées à l'effort produit par les industriels pour fournir des architectures matérielles efficaces ont entraîné l'hégémonie de cette norme : elle a notamment été adoptée par DVB pour succéder au MPEG-2 et par la Blu-ray Disc Association.

Tous les standards vidéo introduits jusqu'ici utilisent les mêmes principes de base, lesquels seront exposés ultérieurement. La norme H.264/AVC FREext, la plus récente et la plus performante à ce jour, est en partie décrite dans la suite de ce rapport car elle constitue le support des travaux réalisés dans le cadre de cette thèse.

Comme principaux concurrents d'H.264/AVC, nous pouvons citer VC-1 [10] standardisé par la SMPTE (*Society of Motion Picture and Television Engineers*) en 2006 et porté par l'entreprise Microsoft. La comparaison entre VC-1 et H.264/AVC a donné lieu à des débats soutenus au sein des communautés de scientifiques et d'utilisateurs. La difficulté pour les départager montre que leurs performances sont plus ou moins équivalentes.

Dans la communauté du logiciel libre, plusieurs standards ouverts ont vu le jour afin de proposer des formats d'encodage vidéo libres de droit. Les plus connus sont Theora sous licence BSD et Dirac soutenu par la BBC et qui a la particularité de reposer sur une transformation de type ondelettes. Malgré ces tentatives, l'utilisation de ces standards reste marginale. En cause, leurs faibles performances en compression qui les placent au niveau de MPEG-2.

Un nouveau venu dans le paysage des solutions de codage vidéo risque de provoquer un bouleversement en proposant une alternative crédible au H.264/AVC : il s'agit de Google et de son format ouvert WebM principalement dédié à la transmission de médias audiovisuels sur le réseau Internet. La partie vidéo du standard WebM est basée sur VP8 et a été publiée sous licence Creative Commons. L'un des points forts de WebM est son système de licence d'exploitation attribuée gratuitement contrairement à H.264/AVC fonctionnant selon un principe de *royalties*.

2.2.2 Les enjeux à venir

Nous l'avons vu en 1.2, les prochaines années vont connaître l'explosion de la consommation de vidéo numérique *via* les réseaux de télécommunication. Face à ces nouveaux défis, les comités MPEG et ITU-T proposent d'élaborer un nouveau standard de codage vidéo appelé HEVC (High Efficiency Video Coding) pour succéder à H.264/AVC. Les objectifs de HEVC sont de réduire le débit de 50% par rapport à H.264/AVC tout en conservant une qualité subjective identique. De plus, la complexité calculatoire du décodeur ne doit pas dépasser plus de trois fois celle d'H.264/AVC, afin de rester en deçà des progrès techniques réalisés concernant la puissance des ordinateurs sur cette période. Les spécifications complètes du standard HEVC sont attendues aux environs de juillet 2012.

2.3 La compression vidéo

Nous nous intéressons ici à un sous-ensemble de méthodes de codage vidéo qui regroupe les plus efficaces et les plus communes. Cette étude sera la base du travail effectué dans les chapitres suivants.

2.3.1 Quelques notions de base

Plusieurs concepts fondamentaux sont partagés par une majorité des codeur-décodeurs (codecs). Nous nous proposons de les présenter succinctement. Tout d'abord, les schémas de codage présentés ici sont de type hybride : le décodeur est partie intégrante du codeur et effectue l'opération inverse de décodage pendant le traitement, ce qui permet d'utiliser la partie du signal décodé pour prédire la suite du signal. Les techniques décrites ci-dessous sont présentées dans l'ordre chronologique de traitement du procédé d'encodage.

2.3.1.1 Sous-échantillonage des composantes chromatiques

Les données brutes des séquences vidéo numérique sont traditionnellement représentées dans l'espace colorimétrique RGB [4], où chaque composante couleur, pour une image de la séquence vidéo, est caractérisée par une matrice d'éléments représentant l'intensité de la couleur. Un pixel est alors défini par l'association des trois couleurs primaires RGB à une position donnée. Le principal inconvénient de cet espace colorimétrique est que chaque composante couleur est d'égale importance, c'est pour cette raison que d'autres systèmes de représentation prenant en compte des aspects de perception visuelle sont souvent préférés, en particulier le modèle YCbCr (ou YUV). La composante Y correspond à la luma, caractérisant l'information de luminance de l'image, tandis que CbCr représentent les deux composantes chromatiques. Les transformations réversibles pour passer de l'espace RGB à YCbCr, définies dans [5], sont données par :

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.114B \\ Cb &= 0.564(B - Y) \\ Cr &= 0.713(R - Y) \end{aligned} \tag{2.3}$$

$$\begin{aligned} R &= Y + 1.402Cr \\ G &= Y - 0.344Cb - 0.714Cr \\ B &= Y + 1.772Cb \end{aligned} \tag{2.4}$$

Le système visuel humain étant moins sensible aux variations de couleurs que de luminosité, ce modèle permet de privilégier la composante Y par rapport aux chrominances. Pour cela, les composantes chromatiques sont sous-échantillonnées pour permettre de réduire la quantité d'information à coder. Le format $4:2:0$ [118] est le plus populaire : la taille des chrominances est alors réduite de moitié verticalement et horizontalement. Ce procédé est illustré sur la figure 2.3, nous pouvons noter que la composante luminance Y représente l'image d'origine exposée en niveaux de gris.

4. Red-Green-Blue

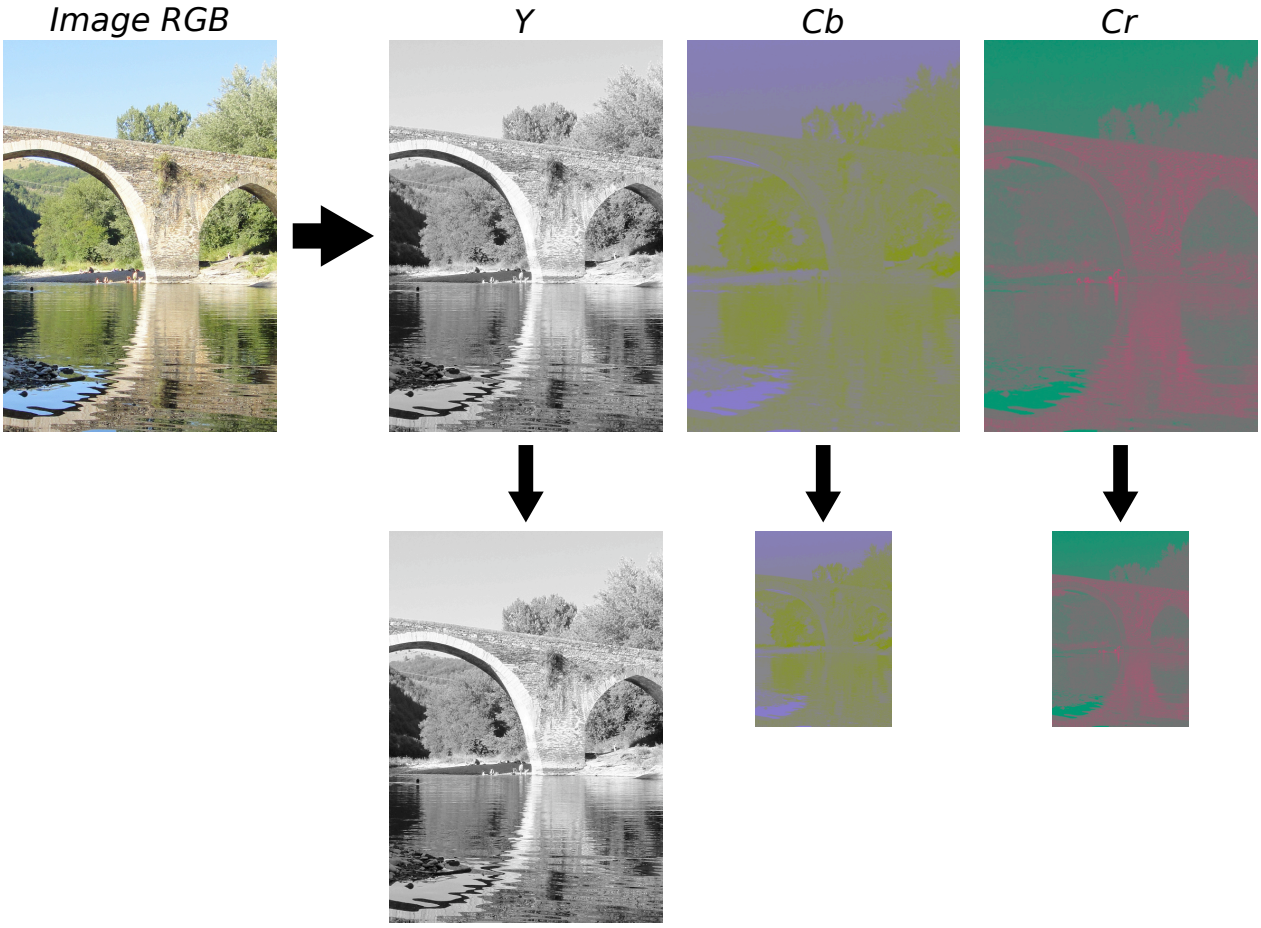


FIGURE 2.3 – Illustration de la transposition d'une image dans l'espace colorimétrique YCbCr suivi du sous-échantillonnage au format 4 :2 :0 des chrominances.

2.3.1.2 Codage par prédiction

Une étape de décorrélation par prédiction est souvent introduite dans les schémas de codage vidéo. Il s'agit en l'occurence de prédire le signal puis de coder l'erreur de prédiction. Plus le procédé de prédiction sera efficace et plus le volume d'information de l'erreur de prédiction sera faible, améliorant ainsi le facteur de compression. La modélisation par un champ de Markov des valeurs successives d'une variable aléatoire x à un instant t nous donne :

$$\tilde{X}(t) = a_1\hat{X}(t-1) + a_2\hat{X}(t-2) + \ldots + a_n\hat{X}(t-n), \tag{2.5}$$

où $\tilde{X}(t)$ est la prédiction de $X(t)$ et où $\hat{X}(t-i)$ est une donnée précédemment reconstruite (nous parlons dans ce cas de prédiction en boucle fermée). La prédiction s'opère par une combinaison linéaire des valeurs précédentes fonction des coefficients a_i. Ces derniers peuvent être fixés et déterminés *a priori*. Il est également possible de les calculer puis de les coder pour chaque variable ou groupe de variables ; nous cherchons alors à minimiser l'erreur de prédiction.

Pour le codage par prédiction des séquences vidéo, chaque image est d'abord partitionnée par blocs de pixels de petites tailles. Les blocs sont alors prédits successivement selon un parcours de gauche à droite et de haut en bas. Deux types de prédiction sont en concurrence :

La prédiction intra-image où un bloc est prédit par interpolation des pixels adjacents appartenant aux blocs de l'image précédemment traités. Cette technique trouve son intérêt dans le fait qu'une image naturelle présente beaucoup de zones homogènes, c'est-à-dire dont la valeur des pixels varie peu.

La prédiction inter-image où un bloc est prédit par recopie d'un bloc d'une image précédemment encodée. Cette technique est très efficace car la fréquence élevée des images (25Hz pour la télévision) montre qu'il y a peu de différence entre deux images successives, notamment si le mouvement des objets dans la scène est pris en compte. Les encodeurs sont dotés d'un estimateur de mouvement et d'un compensateur de mouvement. Ce dernier est chargé de coder la différence entre le mouvement estimé et le mouvement optimal selon le critère débit/distorsion.

Le résidu, ou erreur de prédiction, est calculé par une simple soustraction pixel à pixel entre le bloc d'origine et le bloc prédit.

2.3.1.3 Transformation

Nous avons vu que l'étape de prédiction engendre un résidu qu'il faut coder. Dans le domaine spatial, la pertinence de l'information est difficilement appréciable : tous les pixels ont la même importance du point de vue perceptuel. C'est pour cela que d'autres espaces de représentation permettant de décorréler le signal sont privilégiés. Sous cette perspective, la transformée en cosinus discrète inversible (DCT pour *Discrete Cosinus Transform*) [117] présente un fort pouvoir de compactage de l'énergie : cela s'explique par le fait que les coefficients transformés par DCT sont fortement décorrélés entre eux. Le résultat d'une DCT à deux dimensions sur une matrice $N \times N$ produit une matrice de taille identique dont chaque coefficient correspond à une fréquence du signal. Pour une image naturelle, l'information est essentiellement portée par les coefficients correspondant aux basses fréquences. Un autre avantage est qu'il existe des algorithmes très rapides [147] implémentant la DCT, puisque cette transformée est, du fait de son orthogonalité, séparable. Pour des raisons de

complexité calculatoire et de corrélation entre pixels voisins, la DCT s'opère sur des blocs de petites tailles, souvent 4×4 ou 8×8 pixels.

La transformation du résidu, de type DCT, ne permet pas de comprimer le signal mais y prépare. Elle a surtout l'avantage d'être en assez bonne adéquation avec le système visuel humain : notre acuité visuelle est d'autant plus faible que la fréquence du signal est élevée. En d'autres termes, les distorsions introduites sur les hautes fréquences sont moins perceptibles que lorsqu'elles concernent les basses fréquences. De ce fait, l'information la moins pertinente peut être supprimée, ce qui permet de réduire le débit de codage tout en conservant une qualité de reconstruction acceptable. Ce procédé est appliqué au cours de l'étape de quantification.

Le détail des calculs de la DCT et DCT inverse est présenté ultérieurement dans ce manuscrit.

2.3.1.4 Quantification

En codage d'image numérique, la quantification est le procédé qui permet d'approximer un signal à valeurs dans un ensemble discret de grande taille par des valeurs d'un ensemble discret d'assez petite taille.

La quantification est un procédé *avec perte* puisqu'elle dégrade de façon irréversible le signal. Elle est néanmoins nécessaire pour augmenter le ratio de compression. Pour cela, la quantification scalaire utilisée dans les schémas de codage actuels consiste à attribuer un symbole extrait d'un dictionnaire fini à un coefficient DCT. Plus précisément, le procédé de quantification scalaire est une application Q de $\mathbb{R}$ dans un ensemble fini D, le dictionnaire, composé de N symboles scalaires :

$$Q = \mathbb{R} \to D \text{ avec } D = \{\overline{x_1}, \overline{x_2}, \ldots, \overline{x_N}\}, \tag{2.6}$$

où $\overline{x} = Q(x)$ représente la valeur quantifiée (ou de reconstruction) de x. On dit que $\overline{x}$ est une approximation de x au sens où $\overline{x}$ est sélectionné pour être le plus proche de la valeur d'origine x :

$$Q(x) = \overline{x_i} \text{ ssi } x \in \left[Q_i^-, Q_i^+\right[, \tag{2.7}$$

où Q_i^- et Q_i^+ sont appelés les bornes de l'intervalle de quantification associées à $\overline{x_i}$, ou encore les niveaux de décision. Lorsque les intervalles de quantification sont identiques indépendamment de la valeur de x, soit $\forall i\ Q_i^+ - Q_i^- = c^{ste}$, il s'agit d'un procédé de quantification scalaire uniforme adapté aux sources de distribution uniforme. Pour les même types de sources, les valeurs de reconstruction optimales $\overline{x_i}$ sont obtenus par calcul des centroïdes, soit dans ce cas $\overline{x_i} = \left\lfloor \frac{Q_i^- + Q_i^+}{2} \right\rfloor$.

La taille N du dictionnaire D peut varier : le débit diminue à mesure que le nombre de symboles décroît, provoquant une dégradation plus importante de la qualité de l'image. Ainsi, l'étape de quantification permet d'ajuster le critère débit/distorsion (voir la figure 2.1) en fonction de l'utilisation désirée.

Nous avons vu que, dans un schéma de codage par prédiction, les coefficients résiduels DCT ont un impact perceptuel différent selon la fréquence correspondante. La quantification scalaire permet d'en tirer profit en adaptant le pas de quantification aux coefficients : plus la fréquence associée à un coefficient DCT est importante et plus le pas de quantification utilisé sera important, donc la taille du dictionnaire faible, ceci afin de réduire la quantité d'information à coder.

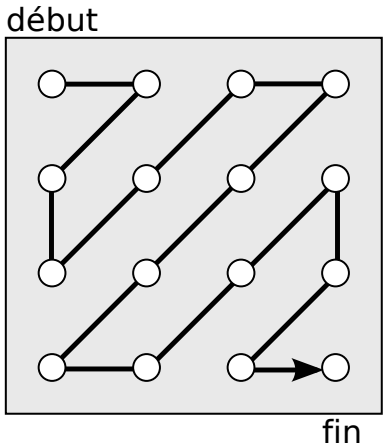


FIGURE 2.4 – Parcours en zig-zag sur un bloc 4×4 de coefficients DCT.

2.3.1.5 Réordonnancement

Nous avons vu que l'étape précédente aboutit à un ensemble de blocs résiduels transformés et quantifiés qu'il faut maintenant coder en langage binaire. Les connaissances *a priori* que nous avons sur la distribution des coefficients DCT vont permettre d'en optimiser le codage. En l'occurence, nous savons que suite à la tranformée DCT et à la quantification, l'information du bloc est portée sur un nombre restreint de coefficients, les autres étant nuls. Dans un premier temps, l'idée consiste à réordonnancer le traitement des coefficients du bloc pour optimiser les plages de coefficients nuls successifs. Les coefficients non nuls, correspondant généralement aux basse-fréquences, sont situés dans le coin supérieur gauche du bloc. Le parcours en zig-zag, illustré sur la figure 2.4, permet de traiter ces coefficients en priorité pour placer les coefficients nuls, correspondant aux hautes-fréquences, en fin de traitement.

Ce réordonnancement est très bien exploité par les algorithmes de codage, par exemple en adoptant une représentation par plage de zéros. Cette technique représente le bloc par un ensemble de paires de nombres (excepté pour le premier coefficient du bloc) : le premier indique le nombre de coefficients nuls qui précèdent ce coefficient non nul courant, le deuxième indique la valeur du coefficient non nul. Un exemple est illustré sur le tableau 2.1. La première ligne indique le positionnement du coefficient dans le bloc 4×4 selon le parcours zig-zag. La deuxième ligne représente la valeur du coefficient DCT et la dernière ligne les symboles à coder pour représenter ce bloc. Le symbole EOB [5] indique que tous les coefficients suivants sont nuls. Nous nous apercevons ici que ce mécanisme permet de réduire le nombre de symboles à coder (12) en lieu et place des 16 coefficients qui composent le bloc. En fait, nous parlons ici de pseudo-codage, dans le sens où la représentation du bloc, bien qu'optimisée, n'est pas exprimée en langage binaire. Pour cela, nous faisons appel au codage entropique présenté ci-dessous.

2.3.1.6 Codage entropique

Enfin, la dernière étape indissociable du processus d'encodage consiste à transformer les données précédemment obtenues en un flux binaire destiné à être stocké ou transmis. Ce procédé de codage

5. End Of Block

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
34	23	7	0	0	5	2	0	0	0	2	0	0	0	0	0

$34, (0, 23), (0, 7), (2, 5), (0, 2), (3, 2)$,EOB

TABLE 2.1 – Exemple de pseudo-codage par plage de zéros après représentation du bloc 4×4 selon le balayage en zig-zag.

sans pertes est basé sur l'utilisation de codes binaires dont la longueur dépend des propriétés statistiques de la source (nous parlons alors de codage VLC [6]). La probabilité d'apparition d'un symbole de la source définit le code binaire qui lui est associé : un code binaire court sera attribué à un symbole fréquent. Les méthodes de codage doivent respecter la contrainte de singularité des codes binaires : ceux-ci doivent être uniques et ne pas préfixer un autre code.

Il existe plusieurs outils de codage entropique, leur objectif étant de se rapprocher de la limite théorique de compression sans pertes, donnée par l'entropie de la source et définie par [127]. L'entropie H d'une variable aléatoire X, à réalisation dans un dictionnaire $\{x_1, \ldots, x_N\}$ et suivant une loi de probabilité $p(x_i)$ pour chaque symbole, est donnée par :

$$H(X) = \sum_{i=1}^{N} -p(x_i) \log_2 p(x_i) \tag{2.8}$$

Parmi les outils les plus courants pour le codage entropique sans perte, nous pouvons citer les méthodes suivantes :

Codage de Huffman [74] basé sur une représentation en arbre binaire des symboles selon leurs probabilités d'apparition. Cette méthode est optimale pour la compression à code de longueur entière d'une source.

Codage Lempel-Ziv [168, 153] basé sur la construction d'un dictionnaire de mots, un mot correspondant à un regroupement de symboles. Ce type de codage est efficace lorsque la source présente des motifs répétitifs de symboles. Il sert notamment de base au format de fichier ZIP et GZIP.

Codage arithmétique [119, 154] qui permet un codage théoriquement optimal. Les deux techniques précédentes de codage représentent les symboles par des codes binaires à longueur entière, ce qui est souvent sous-optimal. Par exemple, pour $p(x_i) > 0.5$, le code le plus court aura une longueur minimale de 1 bit. Le codage arithmétique améliore l'efficacité de codage en convertissant un groupe de symboles en un nombre fractionnaire unique permettant en pratique de coder un symbole sur moins d'un bit.

L'encodeur et le décodeur doivent utiliser la même table d'équivalence entre les symboles et leurs codes binaires associés. Certains standards de codage se basent sur des tables d'équivalence prédéfinies, souvent utiles lorsque la probabilité $p(x_i)$ de chaque symbole est connue et fixe. Par exemple, la probabilité d'apparition d'une lettre de l'alphabet dans un document en français ne varie pas d'un texte à l'autre. A l'inverse, lorsque la distribution de la source n'est pas connue, la table d'équivalence optimale peut être calculée à l'encodeur puis transmise au décodeur. Dans ce cas, le procédé d'encodage nécessite deux passes, la première étant dédiée au calcul des probabilités $p(x_i)$. L'inconvénient de cette méthode est qu'elle n'est pas compatible avec les applications temps réel où les processus

6. Variable-Length Coding

Entier	Représentation binaire	Code exp-Golomb
0	1	1
1	10	010
2	11	011
3	100	00100
4	101	00101
5	110	00110
6	111	00111
7	1000	0001000
8	1001	0001001
...	...	...

TABLE 2.2 – Illustration du codage exponentiel-Golomb.

d'encodage et de décodage s'opèrent instantanément. Enfin, une autre méthode de codage consiste à modifier "à la volée" la table d'équivalence en fonction des symboles codés/décodés. Ce type de codage, appelé codage dynamique, requiert l'utilisation d'algorithmes d'optimisation parfois complexes s'exécutant de façon synchrone à l'encodeur et au décodeur. L'intérêt de cette approche est qu'elle est générique, c'est-à-dire qu'il n'est pas nécessaire d'avoir une connaissance *a priori* des probabilités d'apparition des symboles de la source, celle-ci s'affinant au cours du processus de codage. En revanche, l'apprentissage des lois de probabilité requiert un échantillon suffisamment représentatif de la source. La méthode peut s'avérer inefficace lorsque le nombre de symboles transmis est faible. Pour illustrer ce procédé, un codage de Huffman dynamique est décrit dans [148].

Indépendamment de la méthode de codage, les outils présentés jusqu'ici reposent sur un dictionnaire fini de symboles associés à des probabilités d'apparition. Dans certains cas il peut s'avérer intéressant de disposer d'outils de codage non contraints par un ensemble fini de symboles. Ce concept repose sur un code universel utilisé comme préfixe du code binaire. Le codage exponentiel-Golomb [140], basé sur le code de Golomb [67], est utilisé notamment pour représenter les entiers positifs en code binaire selon le schéma illustré en 2.2. La longueur du code dépend de l'entier : un code court est attribué aux entiers petits. Les codes binaires à longueur variable respectent une construction logique donnée par :

$$[M \text{ zéros}]\,[1]\,[\text{info}] \tag{2.9}$$

où [info] est représenté sur M bits.

Cette technique est utilisée lorsque le codage de Huffman ou arithmétique ne peut être utilisé ; mais l'efficacité en compression est systématiquement moins élevée. Le codage exponentiel-Golomb trouve son intérêt dans un contexte de codage vidéo prédictif où un résidu est statistiquement proche de zéro. Toutefois, dans certains cas où le modèle de prédiction échoue, ce mécanisme permet de coder une valeur quelconque dans un dictionnaire théoriquement de taille infinie.

Dans le cas du codage vidéo, les informations à coder sont de deux types :

La signalisation (ou entête) qui permet de "comprendre" l'organisation du flux binaire décodé : le partitionnement de l'image utilisé, les méthodes de prédiction employées pour chaque bloc ou groupe de blocs, les tables de quantification utilisées, etc. La signalisation est utile pour "*parser*" (ou analyser) la vidéo encodée.

L'information résiduelle correspond au contenu propre à l'image, c'est-à-dire aux données restantes lorsque toutes les méthodes de codage ont été successivement employées : prédiction, transformation et quantification. L'information résiduelle caractérise les données utiles pour le décodeur.

Nous avons vu précédemment comment exploiter la distribution statistique des coefficients résiduels, notamment en adoptant le parcours en zig-zag et un codage par plage. Les informations de signalisation font également l'objet d'optimisations statistiques à plusieurs niveaux pour être représentées par des codes binaires. Nous reviendrons sur ce point au cours de la description technique de la norme H.264/AVC.

2.3.1.7 Filtrage

Le filtrage est un procédé de post-traitement, en général facultatif (excepté pour les normes de codage vidéo les plus récentes), du fait de sa complexité calculatoire ; mais son efficacité est indéniable. Les schémas de codage basé sur une transformée DCT partitionnent l'image en blocs de petites tailles. Une quantification agressive engendre une forte réduction de débit au prix d'une dégradation importante, qui se matérialise par des effets de blocs perceptuellement gênants. Le filtrage, connu sous le nom de *deblocking filter* [88], peut remédier en partie à ce problème en lissant les pixels aux frontières des blocs.

2.3.2 La norme H.264/AVC

2.3.2.1 Introduction

Dans la norme H.264/AVC, seule la partie décodage est normalisée, laissant le soin aux différents encodeurs de sélectionner les paramètres d'encodage. En effet, certains outils de codage, par exemple les méthodes de prédiction, sont en compétition au niveau de l'encodeur. Il revient alors à l'encodeur de choisir le plus approprié selon ses propres critères de choix. Pour ces raisons, les performances des encodeurs matériels ou logiciels ne sont pas identiques et dépendent des choix d'implémentation soumis au compromis d'efficacité en compression/complexité calculatoire. Par exemple, certaines applications de type temps réel nécessitent un encodage à la volée. Pour cela des heuristiques sont utilisées afin de sélectionner le mode d'encodage probablement le plus efficace et ainsi réduire la complexité calculatoire. D'autres applications, par exemple destinées au stockage, ne sont pas contraintes par le temps d'encodage. Dans ce cas, plusieurs passes d'encodage permettent d'optimiser les paramètres d'encodage le plus finement pour une compression maximale.

Le standard définit des profils en fonction des utilisations souhaitées :

Baseline Profile pour les applications disposant de peu de ressources, largement répandu dans les applications destinées aux mobiles ;

Main Profile pour les applications de diffusion et de stockage grand public ;

Extended Profile prévu spécifiquement pour la diffusion en *streaming* ;

High Profile pour répondre aux besoins des applications les plus exigeantes en terme d'efficacité de codage.

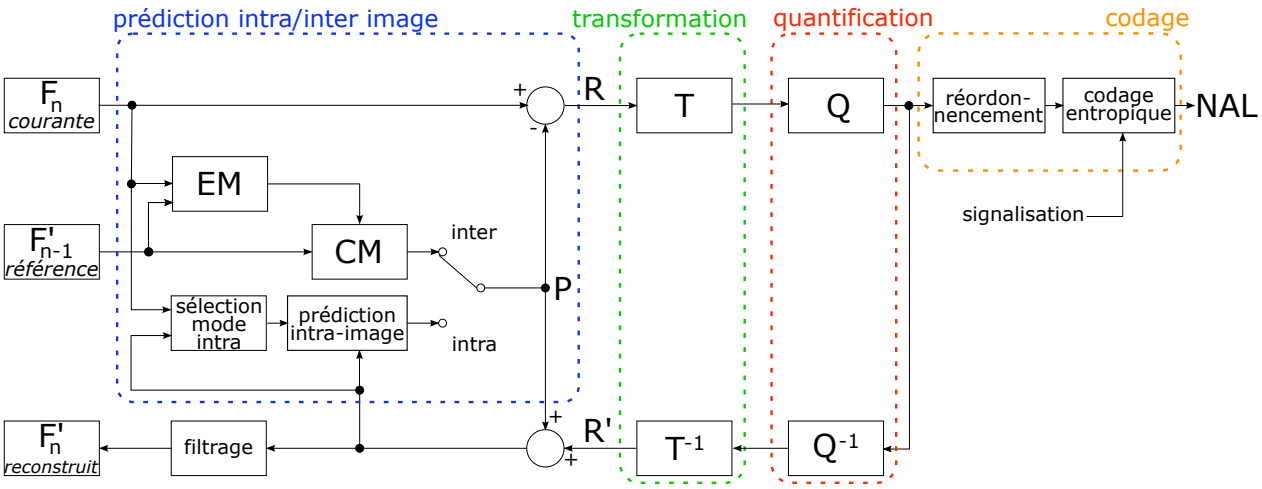


FIGURE 2.5 – Schéma de compression vidéo de la norme H.264/AVC. EM : Estimation de Mouvement. CM : Compensation de Mouvement. $T^{(-1)}$: Transformation (inverse). $Q^{(-1)}$: Quantification (inverse).

Le profil supérieur propose de nouveaux outils de codage plus performants regroupés sous l'extension H.264/AVC FRExt [7] [133, 95, 9] finalisée en juillet 2004. Ce profil s'est rapidement imposé, notamment pour les nouveaux besoins liés à la haute-définition. Il a par exemple été choisi pour l'encodage des disques Blu-ray et pour la radiodiffusion en Europe. Le profil supérieur sera la référence utilisée dans ce manuscrit, c'est pourquoi nous allons l'étudier spécifiquement.

La norme H.264/AVC relève d'une complexité telle que sa description technique complète n'aurait pas sa place dans ce manuscrit. Nous nous focaliserons sur les éléments clés qui permettront de comprendre en détail les implémentations décrites dans les chapitres suivants. L'ouvrage de Richardson [118] fait office de référence pour la description du codage H.264/AVC. Le schéma global de codage de la norme H.264/AVC est présenté sur la figure 2.5.

La norme H.264/AVC est basée sur un partitionnement par macro-bloc de taille 16×16 pixels pour la composante luma et 8×8 pour les composantes chromatiques (selon une représentation de type $4:2:0$, voir 2.3.1.1). Chaque macro-bloc peut lui même être décomposé en blocs de tailles et de formes différentes. Les macro-blocs sont étiquetés selon un type de codage qui va conditionner la méthode de prédiction utilisée :

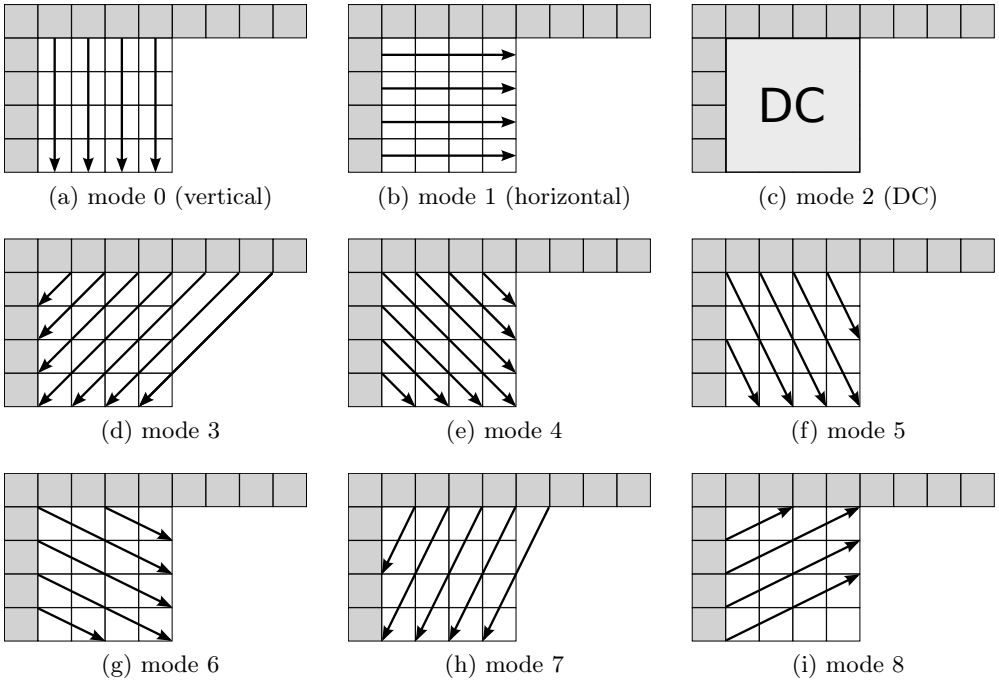
macro-bloc *I* pour les macro-blocs prédits par une méthode intra-image, typiquement une interpolation à partir des pixels des blocs adjacents préalablement encodés ;

macro-bloc *P* pour les macro-blocs prédits par une méthode inter-image à partir des images de référence préalablement encodées, en utilisant une estimation puis compensation de mouvement ;

macro-bloc B comme pour les macro-blocs *P*, à la différence près que les images de référence peuvent être des images futures selon l'ordre temporel de la séquence (prédiction bi-prédictive).

Les travaux présentés dans ce manuscrit sont des méthodes de modélisation des images par analyse/synthèse de texture et par restauration dans un contexte de prédiction intra-image. Bien que nos recherches puissent être étendues au codage inter-image, les contributions présentées s'appuient sur

7. *Fidelity Range Extensions*



(a) mode 0 (vertical) (b) mode 1 (horizontal) (c) mode 2 (DC)

(d) mode 3 (e) mode 4 (f) mode 5

(g) mode 6 (h) mode 7 (i) mode 8

FIGURE 2.6 – Neufs modes de prédiction intra-image pour les blocs luma 4×4 pixels. En gris les pixels des blocs adjacents préalablement encodés.

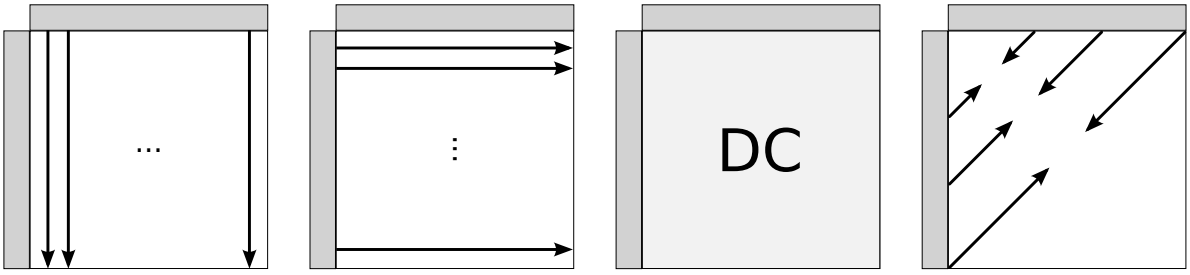
le codage de macro-blocs *I*. C'est pourquoi nous allons présenter en détail les méthodes de prédiction intra-image de la norme H.264/AVC FREext.

2.3.2.2 Prédiction intra-image

La prédiction intra-image dans H.264/AVC FREext fonctionne par interpolation des pixels voisins reconstruits en fonction d'une direction prédéfinie. Le parcours des blocs s'opérant de gauche à droite et de haut en bas (*raster scan*), les pixels voisins reconstruits appartiennent aux blocs préalablement codés donc situés au dessus et à gauche du bloc courant. Quelques subtilités différencient les modes de prédiction selon le partitionnement utilisé. Pour la composante luma, la norme décrit trois possibilités de partitionnement pour un macro-bloc : seize blocs de taille 4×4, quatre blocs de taille 8×8 ou un bloc de taille 16×16 pixels.

Pour le partitionnement par blocs 4×4 et 8×8, neuf modes de prédiction intra-image co-existent : huit selon une interpolation directionnelle des pixels voisins plus un mode DC où une valeur unique est attribuée aux pixels du bloc courant correspondant à la moyenne de tous les pixels connus voisins. Les neuf modes de prédiction intra-image pour les blocs 4×4 sont illustrés sur la figure 2.6. Par extension, les modes de prédiction sont identiques pour les blocs luma 8×8 pixels.

Pour le partitionnement par bloc 16×16 pixels, seuls quatre modes de prédiction intra-image sont utilisés par interpolation des pixels adjacents du macro-bloc situé à gauche et/ou supérieur (figure 2.7). De même, ces quatre modes sont utilisés pour la prédiction de chaque composante chromatique 8×8 pixels qui ne bénéficient pas de partitionnement au niveau du macro-bloc.



(a) Mode 0 (vertical) (b) Mode 1 (horizontal) (c) Mode 2 (DC) (d) Mode 3 (diagonal)

FIGURE 2.7 – Quatre modes de prédiction intra-image pour les macro-blocs luma 16×16 pixels. En gris les pixels des blocs adjacents préalablement encodés.

Suite à cette étape, un résidu est obtenu en soustrayant le bloc d'origine avec le bloc prédit. Ce résidu est alors transformé par une DCT puis quantifié.

2.3.2.3 Transformation et quantification

La norme H.264/AVC FRExt est basée sur une transformée DCT modifiée qui offre l'avantage d'opérer sur des nombres entiers et de pouvoir être implémentée en n'utilisant que des opérations simples : addition, soustraction et décalage de bits. La DCT est activée pour des blocs 4×4 ou 8×8, elle s'adapte de ce fait aux partitionnements 4×4 et 8×8 utilisés au cours du procédé de prédiction intra-image (2.6). Pour la prédiction intra-image s'opérant au niveau macro-bloc (2.7), celui-ci est sub-divisé en seize blocs luma 4×4 et subit deux transformations successives. Dans un premier temps, la DCT est utilisée sur chacun des blocs 4×4 ainsi formé. Ensuite, la transformée de Hadamard 4×4 est appliquée sur le bloc constitué de tous les coefficients DC du macro-bloc obtenus par la DCT. Le même mécanisme de transformation à deux étages est utilisé pour chaque composante chroma.

La transformée DCT 4×4 et 8×8 est factorisée pour réduire le nombre d'opérations nécessaires ([93]). Une matrice α_N de taille $N \times N$ (avec $N = 4$ ou $N = 8$) est obtenue à partir d'une matrice de pixels u_N selon :

$$\alpha_N = \left[(\mathrm{A}_N)\ u_N\ (\mathrm{A}_N)^T \right] \otimes \mathrm{E} \tag{2.10}$$

où A_N et $(A_N)^T$ sont le corps de la transformée DCT, tandis que E est une matrice de mise à l'échelle. L'opérateur $\otimes$ correspond à une multiplication terme à terme entre deux matrices. Dans H.264/AVC, les matrices A_4 et A_8 sont définies par :

$$\mathrm{A}_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \quad \mathrm{A}_8 = \begin{bmatrix} 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 12 & 10 & 6 & 3 & -3 & -6 & -10 & -12 \\ 8 & 4 & -4 & -8 & -8 & -4 & 4 & 8 \\ 10 & -3 & -12 & -6 & 6 & 12 & 3 & -10 \\ 8 & -8 & -8 & 8 & 8 & -8 & -8 & 8 \\ 6 & -12 & 3 & 10 & -10 & -3 & 12 & -6 \\ 4 & -8 & 8 & -4 & -4 & 8 & -8 & 4 \\ 3 & -6 & 10 & -12 & 12 & -10 & 6 & -3 \end{bmatrix} \tag{2.11}$$

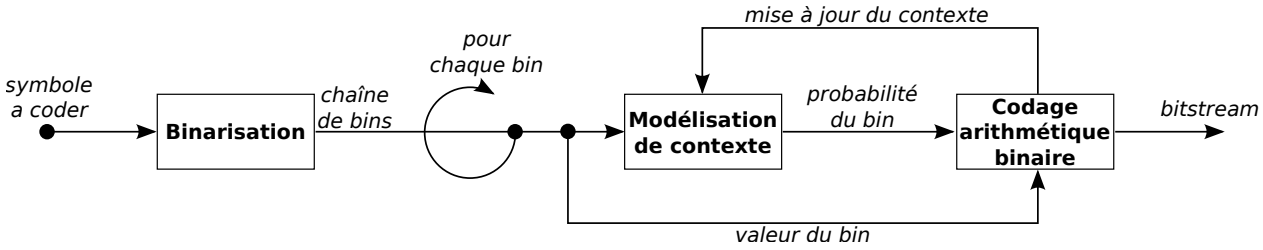


FIGURE 2.8 – Diagramme de l'encodeur CABAC utilisé dans H.264/AVC FREXt.

L'intérêt de cette approche est d'obtenir des matrices A_4 et A_8 entières facilitant l'implémentation de la DCT. La matrice E de mise à l'échelle est ensuite combinée à la matrice opérant la quantification scalaire pour mutualiser les traitements.

De façon classique, la quantification scalaire est réalisée par l'opération :

$$q_{ij} = arrondi\left(\frac{\alpha_{ij}}{Q_{step}}\right) \tag{2.12}$$

où α_{ij} correspond à l'élément de la matrice α positionné en (i, j). Le paramètre Q_{step} correspond au pas de quantification choisi parmi un total de 52 valeurs prédéfinies dans le standard H.264/AVC, ce qui permettent d'ajuster le critère débit/distorsion. Ainsi, le paramètre Q_{step} prend également en compte la matrice de mise à l'échelle E pour ne plus réaliser qu'une seule et même opération. Le mécanisme permettant d'unir les procédés de transformation et de quantification est détaillé dans [118].

2.3.2.4 Codage entropique

Le codage entropique est un procédé dit "sans-perte", qui représente l'information sous forme binaire et structurée afin d'être compréhensible par le décodeur. Dans H.264/AVC FREXt, le précédent codeur entropique CAVLC[8] [26] est remplacé par le plus performant CABAC[9] [94]. Un gain en compression entre 10 et 15% est constaté grâce à l'emploi de CABAC par rapport à son prédécesseur, même si cela se traduit par une augmentation notable de la complexité et des ressources mémoires nécessaires. Nous allons revenir plus en détail sur ce dernier, dont le schéma global est illustré sur la figure 2.8.

Le procédé de binarisation, ou d'expression sous forme binaire, consiste à transformer chaque symbole en chaînage de *bin*, où un *bin* correspond à un symbole binaire. Cette étape est indispensable pour la suite puisque CABAC utilise un codeur arithmétique binaire, c'est-à-dire qu'il n'est capable de coder que des $'0'$ ou $'1'$. Pour cela, différents types de table d'équivalence entre un symbole en entrée, qui caractérise un élément de syntaxe, et un chaînage binaire, sont utilisés. En fonction de l'élément de syntaxe, nous pouvons utiliser le code unaire, unaire tronqué, Exp-Golomb (vu en 2.3.1.6) ou à longueur fixe [94]. L'utilisation de ces dictionnaires fixes rend le procédé rapide et efficace : il permet d'approcher la distribution de la source par un code binaire adapté.

La modélisation de contexte permet d'estimer la distribution d'un *bin* en fonction des symboles précédemment encodés, avec l'objectif de proposer une probabilité binaire la plus fiable possible

8. *Context-Adaptive Variable-Length Coding (CAVLC)*
9. *Context-Adaptive Binary Arithmetic Coding (CABAC)*

au codeur arithmétique. Chaque *bin* x à coder possède un ensemble de contextes associés $\mathbf{C} = \{0, \ldots, C-1\}$. Le contexte choisi pour coder un *bin* dépend d'un ensemble $\mathbf{T}$ composé des symboles préalablement codés. La fonction de modélisation du contexte est alors donnée par $F : \mathbf{T} \rightarrow \mathbf{C}$. Ainsi, la probabilité conditionnelle $p(x|F(z))$ associée à un symbole à coder x est fonction du contexte obtenu suite à l'étude des symboles préalablement codés $z \in \mathbf{T}$. Ensuite, un codeur arithmétique binaire est utilisé pour encoder le *bin* x avec le modèle de probabilité $p(x|F(z))$. Suite à l'encodage de x, le modèle de probabilité du contexte est mis à jour afin d'affiner la distribution statistique de la source.

En pratique, la modélisation du contexte est souvent fonction des deux blocs voisins préalablement codés. Par exemple, le contexte utilisé pour coder un vecteur de mouvement (MVD [10]) dépend des distances L_1 des MVD des deux blocs voisins.

2.3.2.5 Conclusion

Nous avons vu quelques spécificités du standard H.264/AVC à chaque étape du codage : prédiction, transformation du résidu, quantification et codage entropique. Plus précisément, nous nous sommes intéressés au modèle de prédiction intra-image qui interpole les pixels adjacents d'un bloc selon une direction ou selon un calcul de moyenne. La prédiction intra-image utilisée dans H.264/AVC est un outil puissant car elle permet de décorréler en grande partie les pixels d'un bloc. Elle part simplement du principe qu'un bloc ressemble à son voisinage direct. Cependant, nous nous apercevons que la prédiction par lissage des pixels voisins n'est pas toujours optimale, certains types de contenus complexes ne sont pas adaptés à cette technique. C'est d'autant plus vrai lorsque la prédiction intra-image concerne des blocs de grande taille plus à même de représenter une structure d'image complexe. C'est ce que nous proposons d'étudier dans la partie suivante dédiée à l'analyse/synthèse de texture.

2.3.3 La norme MPEG-4 *Visual*

La norme MPEG-4 est plus ambitieuse que son prédécesseur MPEG-2 et intègre de nouvelles applications qui vont bien au-delà du codage vidéo. Nous nous intéresserons à un sous-ensemble d'outils mis à disposition dans MPEG-4 *Visual* [11] en rapport à la notion d'objets vidéo. Nous parlons alors de mode *objet* et *sprite* qui se différencie du mode "rectangulaire" qui correspond à une approche classique du procédé d'encodage selon un partitionnement de l'image en macro-blocs. Nous présentons ici les concepts sous-jacents.

2.3.3.1 Mode *objets*

Dans ce mode, une séquence vidéo est caractérisée par une collection d'objets vidéo. La superposition de ces objets vidéo permet de reconstruire la séquence d'origine, mais d'autres d'utilisations sont rendues possibles, comme la visualisation d'un objet spécifique, la suppression d'un objet etc. La forme des objets vidéos est quelconque et peut évoluer dans le temps. Dans un contexte de compression, l'idée sous-jacente est d'identifier et de segmenter les objets indépendants d'une séquence qui ont une texture et un mouvement propre. Il faut bien noter ici que la norme MPEG-4 ne fournit pas

10. *Motion Vector Difference (MVD)*
11. ou MPEG-4 Part 2

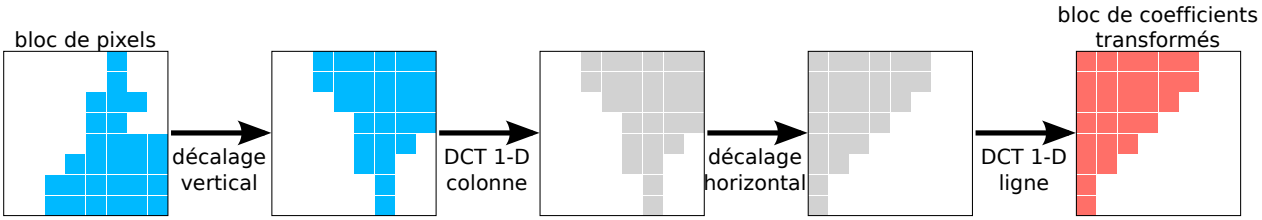


FIGURE 2.9 – Illustration de la SA-DCT. En bleu les pixels du bloc à coder, en rouge les coefficients transformés obtenus.

d'outils de segmentation ; elle offre uniquement un ensemble d'outils pour représenter, manipuler et coder de façon efficace les objets d'une séquence. Un état de l'art sur les méthodes de segmentation spatio-temporelle, appliquées dans un contexte de codage vidéo, est disponible dans l'article de Salembier et Marques [123].

Pour caractériser un objet dans la scène, MPEG-4 définit la notion d'objets vidéo de plan, ou VOP [12]. Un VOP correspond au plus petit rectangle englobant l'objet et sa taille peut évoluer dans le temps. Il est donc nécessaire, pour chaque VOP, de coder sa texture et sa forme.

Puisqu'il s'agit d'une segmentation basée pixel, la forme de l'objet segmenté doit être spécifiée pour tous les macro-blocs frontaliers du VOP. Pour cela, la norme MPEG-4 code un masque binaire appelé BAB [13]. Cependant, plusieurs techniques d'estimation/compensation du BAB existent et se basent sur la segmentation de l'objet dans une image de référence.

L'information de texture, ou de résidu de texture lorsqu'il y a eu estimation de mouvement, subit une transformation DCT. Là encore, un traitement spécifique est appliqué aux blocs frontaliers de l'objet du fait de leurs configurations particulières. La technique utilisée, dite SA-DCT [14][128], est basée sur des transformations DCT mono-dimensionnelles appliquées verticalement puis horizontalement. Dans un premier temps, le sous-ensemble des pixels du bloc à coder est décalé colonne par colonne vers le haut avant de subir une transformation verticale. Ensuite, les mêmes opérations de décalage puis de transformation des coefficients sont appliquées horizontalement, comme illustré sur la figure 2.9. L'opération inverse est effectuée au décodage pour reconstruire le sous-ensemble des pixels du bloc. La SA-DCT offre l'avantage d'obtenir autant de coefficients transformés que de pixels dans le bloc, permettant ainsi de réduire la quantité d'information à coder.

2.3.3.2 Mode *sprite*

Dans la norme MPEG-4, le terme *sprite* désigne la notion d'image panoramique ou mosaïque correspondant au fond statique d'une séquence vidéo. Cette mosaïque est obtenue après analyse de la séquence vidéo complète. Les mouvements de caméra sont ainsi pris en compte pendant le plan séquence (comme le *travelling*). L'intérêt de cette approche est que le fond, la partie statique de la séquence, n'évolue pas au cours du temps. Il est alors possible d'encoder qu'une seule et unique fois l'image panoramique pour tout le plan séquence, ce qui supprime les redondances d'information entre les images successives. Il ne reste plus qu'à coder la position de chaque image dans la mosaïque pour reconstruire le fond de la séquence d'origine. Ensuite, les objets dynamiques de la scène sont insérés pour reconstruire la vidéo d'origine. Le procédé est illustré sur la figure 2.10.

12. *Video Object Plane*
13. *Binary Alpha Block*
14. *Shape-Adaptive* DCT

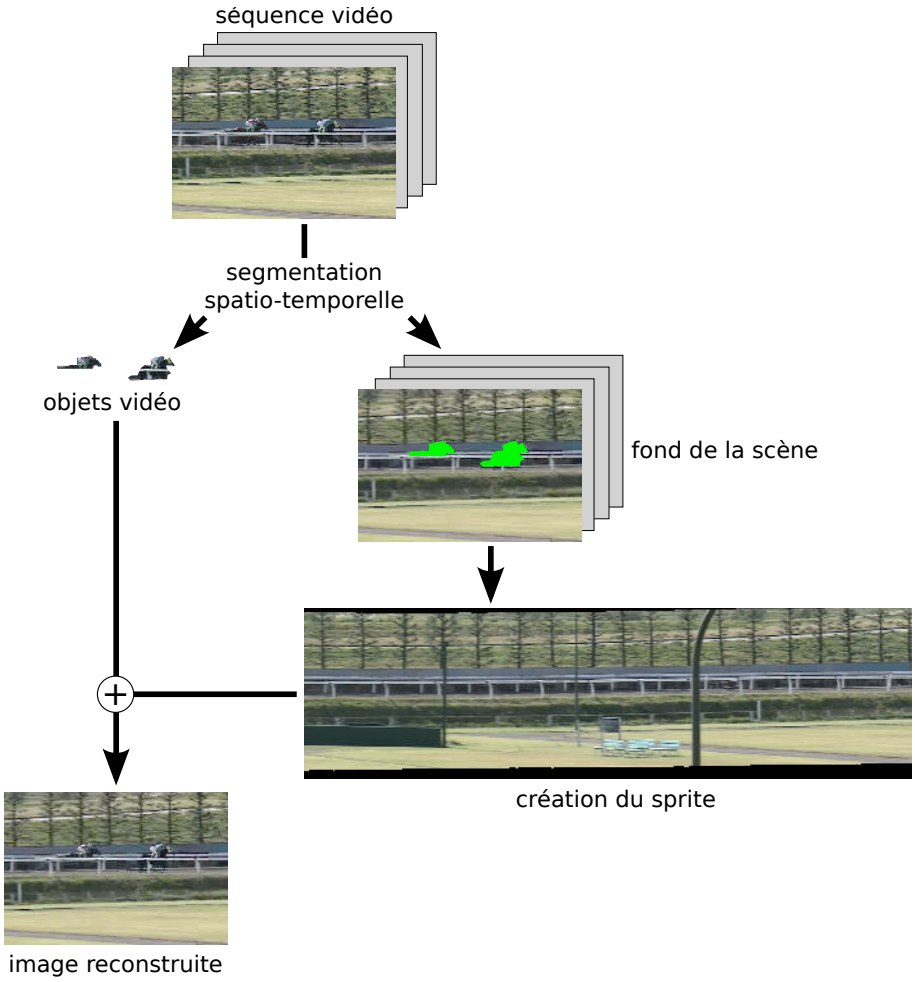


FIGURE 2.10 – Illustration des modes de codage *objets* et *sprite* dans MPEG-4 *Visual*.

2.3.3.3 Conclusion

Les deux modes *objets* et *sprite* de MPEG-4 *Visual* présentent une approche intéressante qui aurait pu être la base des travaux réalisés au cours de cette thèse. En effet, la représentation basée contenu d'une séquence peut servir de support à un codage efficace par analyse/synthèse des objets. Cependant, la segmentation spatio-temporelle est une opération complexe et souvent peu fiable qui peut alimenter un sujet de thèse à part entière, et prend en considération des aspects temporels en dehors de notre champ d'étude. Nous constatons d'ailleurs que les différents travaux basés sur cette approche offrent des performances en compression intéressantes seulement pour un type restreint de séquences vidéo. En général, il s'agit d'un déplacement simple de caméra dans le plan ou d'une scène dont les objets sont facilement segmentables dans l'espace et dans le temps (sans variation d'illumination par exemple), c'est rarement le cas pour des séquences vidéo dites "naturelles". Il est donc difficile d'aboutir à un outil générique fiable pour coder efficacement tout type de séquence. De plus, le facteur de compression de MPEG-4 *Visual* est loin d'être optimal à cause de ses difficultés d'implémentation par rapport à H.264/AVC FREext présenté précédemment. C'est pour ces raisons que les travaux de cette thèse se sont basés sur l'ajout de nouveaux outils dans H.264/AVC, avec une représentation vidéo classique par macro-bloc.

Deuxième partie

Analyse/synthèse de texture

CHAPITRE 3

Codage vidéo par analyse/synthèse de texture : un état de l'art

Sommaire

3.1 Introduction

Cet état de l'art dresse un panorama non exhaustif des méthodes d'analyse et de synthèse utilisées pour le codage des images fixes ou animées. Nous nous intéressons à la synthèse des régions de texture mais aussi de structures (contours...). L'idée sous-jacente de telles méthodes est de remplacer une partie naturelle de l'image (bloc, région, objet) par une partie synthétisée, ce qui peut s'avérer intéressant si la représentation synthétique est "suffisante" (au sens de la qualité perçue) et moins coûteuse (en termes de débit) que le codage utilisant une méthode classique. Dans ce chapitre, nous allons nous focaliser sur quatre domaines d'application du traitement d'images numériques :

L'analyse d'images : il s'agit de décrire partiellement ou complètement une scène à partir de l'image observée (objets présents, dimensions, position dans l'espace,...). Classiquement, ce processus d'analyse se décompose en prétraitement et extraction des traits caractéristiques, puis classification et reconnaissance de formes avant de décrire (et éventuellement interpréter) l'image.

La synthèse d'images : le but de la synthèse est de reconstruire une image qui ressemble à l'image simulée à partir d'une description de la scène, des objets qui la composent, etc. Cette scène reconstruite peut être ressemblante à la réalité ou imaginaire.

La restauration ou amélioration des images : il s'agit de modifier une image de manière "indétectable". Une région manquante ou détériorée est "remplie", le plus souvent par des informations synthétisées grâce à l'étude de son voisinage. Ce procédé, appelé *inpainting*, est en grande partie issu des recherches antérieures sur l'analyse/synthèse des images numériques.

Le codage avec compression : au regard des résultats très prometteurs obtenus par les algorithmes de restauration, plusieurs recherches récentes ont intégré ces algorithmes dans le schéma de compression des images. Le principe repose sur la suppression volontaire de certaines parties de l'image choisies judicieusement, et qu'il sera ensuite possible de reconstruire grâce au voisinage spatial et/ou aux informations supplémentaires codées qui décrivent la partie manquante.

Le domaine qui nous intéresse est à la convergence de ces quatre champs applicatifs. L'utilisation de méthodes de synthèse pour représenter des portions de l'image au lieu de les coder par des méthodes classiques suppose d'avoir au préalable effectué une analyse pour savoir quelles parties de l'image nécessitent d'être codées classiquement et lesquelles doivent être synthétisées. Par ailleurs le choix de représenter une partie de cette image par synthèse (plutôt que codage) constitue bien une forme de restauration. Les techniques utilisées appartiennent donc à tous ces domaines et en mixent les techniques : modèles statistiques issus des méthodes de synthèse les plus classiques (image modélisée par un champ de Markov), méthodes heuristiques plus légères (de type "copier-coller"), méthodes de restauration basées sur des équations de diffusion ou de recouvrements d'échantillons (*patchs*)... Ces méthodes ne donnent pas à elles seules de résultats de codage satisfaisants : elles sont la plupart du temps couplées à des méthodes de codage classiques (codage hybride) avec lesquelles elles entrent en coopération ou en compétition.

Dans ce chapitre, nous nous focaliserons sur les méthodes de synthèse et de restauration d'image qui ont inspiré les algorithmes de compression d'image et de vidéo. La première partie est consacrée à la définition généraliste de "texture".

3.2 Qu'est ce qu'une texture ?

La notion de "texture" en imagerie est un concept abstrait qui peut faire référence à différentes caractéristiques d'une image. Bien qu'il y ait beaucoup de phénomènes naturels décrits comme des textures, il n'y a pas de définition universellement acceptée. Voici quelques définitions intéressantes trouvées dans la littérature :

> *"The term texture generally refers to repetition of basic texture elements called texels. The texel contains several pixels, whose placement could be periodic, quasi-periodic or random. Natural textures are generally random, whereas artificial textures are often deterministic*

> *or periodic. Texture may be coarse, fine, smooth, granulated, rippled, regular, irregular, or linear."* par Jain dans [75]
>
> *"Textures are homogeneous patterns or spatial arrangements of pixels that regional intensity or color alone does not sufficiently describe. As such, textures have statistical properties, structural properties, or both. They may consist of the structured and/or random placement of elements, but also may be without fundamental subunits."* par Smith dans [130]
>
> *"Textures usually can be described informally as the output of some physical process wherein local structure is repeated seemingly at random. Two texture patches are considered to be the same if they appear to have been generated by the same process."* par De Bonet dans [30]

L'approche que nous proposons consiste à décrire une image naturelle comme une composition de contours et de textures : les contours déterminent la structure des objets alors que la texture définit le remplissage des structures. Dans le domaine de la vision, une texture sous-entend un signal "homogène" constitué d'éléments répétés souvent aléatoirement (couleur, position, orientation, etc.), c'est-à-dire un signal susceptible d'être décrit statistiquement. L'interprétation d'une texture prend également en compte une notion de distance et d'angle d'observation, ainsi que de degré d'attention visuelle. De ce point de vue, tout ce qui compose l'univers peut être considéré comme une texture, à partir du moment où les conditions de visualisation sont adéquates, d'un assemblage régulier d'atomes jusqu'à l'observation d'une galaxie.

Les textures sont souvent cataloguées en deux catégories :

Les textures déterministes (ou périodique ou régulière ou rigides) sont caractérisées par une répartition spatiale régulière d'un motif géométrique constant. La description du motif élémentaire, les dimensions du réseau et son orientation suffisent à décrire la texture. Cette définition ne convient qu'à des textures parfaitement régulières que l'on ne rencontre que très rarement dans la réalité des images naturelles.

Les textures stochastiques (ou aléatoires ou non-rigides), à l'inverse, sont caractérisées par des motifs différents appartenant à une population dont seules les propriétés statistiques sont définies et dont la répartition spatiale suit également une grille irrégulière. Il est alors difficile d'isoler un motif de base.

Cependant, beaucoup de textures naturelles sont situées entre ces deux extrêmes. La figure 3.1 illustre quelques-unes de ces textures.

Par extension à la vidéo, une nouvelle nomenclature différencie les textures stationnaires dans le temps (exemple : mur de brique) aux textures temporellement aléatoires, c'est-à-dire les textures vouées à se transformer continuellement (surface de l'eau, flamme...).

De par leurs descriptions très hétéroclites, l'analyse et la classification automatique de texture est un problème difficile. L'analyse de texture est en général regroupé en deux grandes familles : les techniques dites statistiques et les techniques spectrales.

A propos des méthodes statistiques, les statistiques d'ordre premier ou deuxième sont couramment utilisées pour classifier des textures basiques (sans structures). Nous pouvons également citer les méthodes LBP [1] [107, 112] qui appliquent une étiquette à chaque pixel en fonction du masque binaire obtenu après analyse du voisinage local. Ce type de méthode a l'avantage d'être peu complexe. Enfin,

1. *Local Binary Patterns*

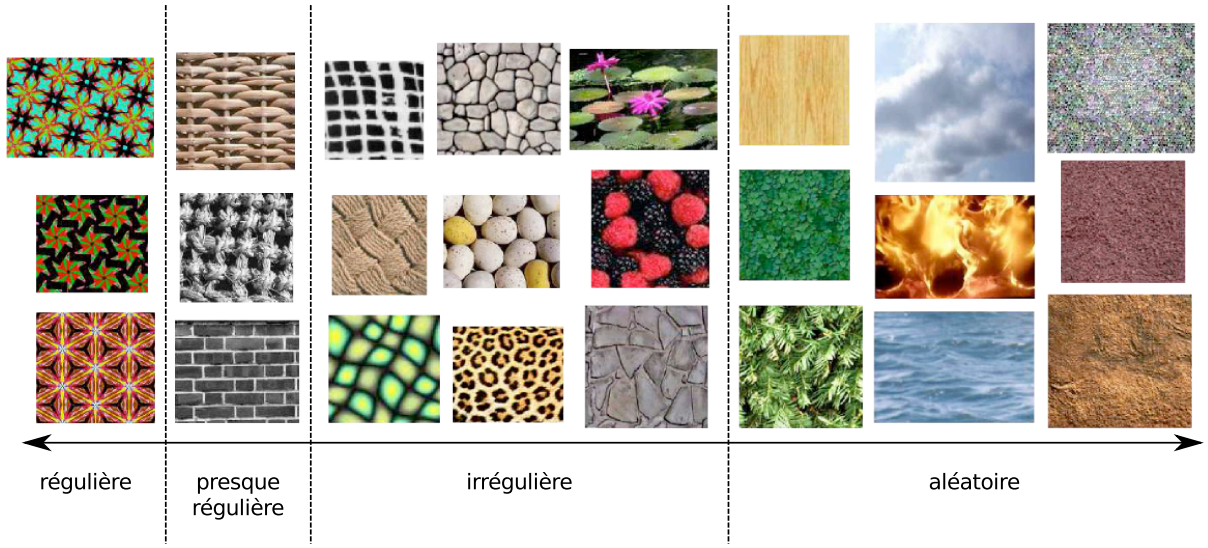


FIGURE 3.1 – Exemple de textures, triées selon la régularité de leurs structures (source : [87]).

les modèles de champs aléatoires [30] représentent chaque pixel comme une valeur engendrée par une loi de distribution de probabilité selon sa position dans l'image et donnée par les autres pixels.

Le plus souvent, les méthodes spectrales [60, 21, 115] sont basées sur une transformée de type ondelette. La décomposition spatio-fréquentielle offre un bon support pour caractériser les singularités d'un signal.

3.3 La synthèse de texture

L'objectif de la synthèse de texture probabiliste peut être formulée comme suit : créer une nouvelle image à partir d'un exemple de texture de telle sorte que la nouvelle image soit suffisamment différente de l'originale mais qu'elle semble produite par le même processus stochastique sous-jacent ([29]). Comme vu précédemment, il existe plusieurs types de texture (déterministes, aléatoires). De ce fait, la littérature est composée de nombreuses méthodes de synthèse de texture, chacune étant souvent adaptée à une catégorie spécifique de texture. Nous recensons deux types de méthodes de synthèse : paramétriques et non paramétriques.

3.3.1 Les méthodes paramétriques

Les méthodes paramétriques utilisent des paramètres statistiques pour décrire une texture. Elles se décomposent en deux phases : tout d'abord la texture en entrée est "analysée" pour en extraire des paramètres statistiques, qui sont ensuite appliqués sur un bruit blanc pour "synthétiser" une texture visuellement proche de l'original. Toute la question réside dans le choix de la représentation de la texture d'entrée pour une analyse statistique efficace et le choix des paramètres statistiques utilisés pour la caractériser au mieux.

Suivant la modalité des images à étudier, la signature la plus discriminante de la texture est à rechercher soit dans des méthodes qui exploitent directement les propriétés statistiques de la texture

(matrices de co-occurrence, fonction d'autocorrélation, modèle de Markov...), soit dans des méthodes qui exploitent les propriétés statistiques à partir d'un plan transformé dans lequel la texture est réécrite (densité spectrale, méthode des extrema locaux, méthodes de transformation de Fourier, décomposition pyramidale...). La décomposition pyramidale, basée sur une représentation hiérarchique de l'image, est particulièrement bien adaptée à ce contexte d'utilisation, puisqu'elle permet de capturer différents traits caractéristiques d'une texture en fonction du niveau de décomposition. Parmi les décompositions pyramidales les plus courantes, deux sont particulièrement privilégiées en analyse/synthèse de texture :

la pyramide laplacienne [32] est construite par la différence entre deux niveaux d'une pyramide gaussienne. Plus précisément, un niveau k de la pyramide laplacienne est le résultat d'une soustraction d'un niveau k d'une pyramide gaussienne et de sa version estimée par expansion du niveau supérieur $k-1$. Le procédé est récursif sur la sous-bande basse-fréquence pour obtenir un nombre de décompositions variable.

la pyramide "steerable" [61, 129] est une décomposition linéaire multi-échelles et multi-orientations par ondelettes du signal. Ce type d'ondelettes possède plusieurs propriétés supplémentaires par rapport aux ondelettes orthogonales traditionnelles, notamment en ce qui concerne l'invariance en translation et en rotation, et de robustesse dû à sa redondance. Elle est utilisée dans le contexte de la synthèse d'image car elle permet de saisir des traits caractéristiques de texture concernant son orientation et possède de très bonnes propriétés de reconstruction, contrairement aux filtres de Gabor ([114]).

Les paramètres statistiques de texture sont analysés à chaque niveau de la pyramide (coefficients d'asymétrie, d'aplatissement, variance, autocorrélation, corrélation croisée, histogramme de couleurs etc. des sous-bandes). La procédure de synthèse est récursive : le procédé est initié par une image de bruit blanc Gaussien décomposée sur une pyramide. Puis une procédure "*coarse to fine*" impose des statistiques sur les sous-bandes à chaque étage de la pyramide.

David Heeger et James Bergen furent parmi les premiers à présenter un procédé de synthèse basé sur une décomposition pyramidale du signal [73]. Leur approche part du principe que les propriétés d'une texture peuvent être mesurées statistiquement à partir des réponses à des filtres linéaires adaptés. Pour cela, deux types de décomposition sont employées : la pyramide Laplacienne et la pyramide "steerable". Pour chaque sous-bande, un histogramme est généré en fonction de la valeur de ses coefficients. La synthèse va ensuite consister à faire "correspondre" les histogrammes d'un bruit blanc aux histogrammes de la texture d'entrée. La reconstruction du signal dans le domaine spatial fournira une texture visuellement similaire mais différente de la texture d'origine. Bien que les résultats ne soient pas satisfaisants pour tous les "types" de textures, et notamment pour les textures composées de structure, ces travaux sont très couramment cités dans le domaine de la synthèse. Par exemple, sur la Fig.3.2, nous observons l'efficacité de la méthode sur des textures aléatoires (ligne du haut) contrairement aux textures composées de structures (deuxième ligne).

Les travaux de De Bonet [29] et Portilla et al. [115] ont étendu le concept en affinant le nombre et la nature des paramètres statistiques utilisés, dans le but d'améliorer la prise en compte des structures durant le processus d'analyse/synthèse. La Fig. 3.3 illustre les résultats obtenus par l'algorithme de synthèse de texture proposé par De Bonet. Nous constatons que la méthode donne de meilleurs résultats par rapport à la méthode de Heeger (voir Fig. 3.2) notamment pour les textures déterministes. Bar-Joseph et al. [19] proposent un algorithme de mixage de textures, basé sur les travaux de De Bonet [29], dont le but est de générer une image à partir de plusieurs échantillons de textures différentes.

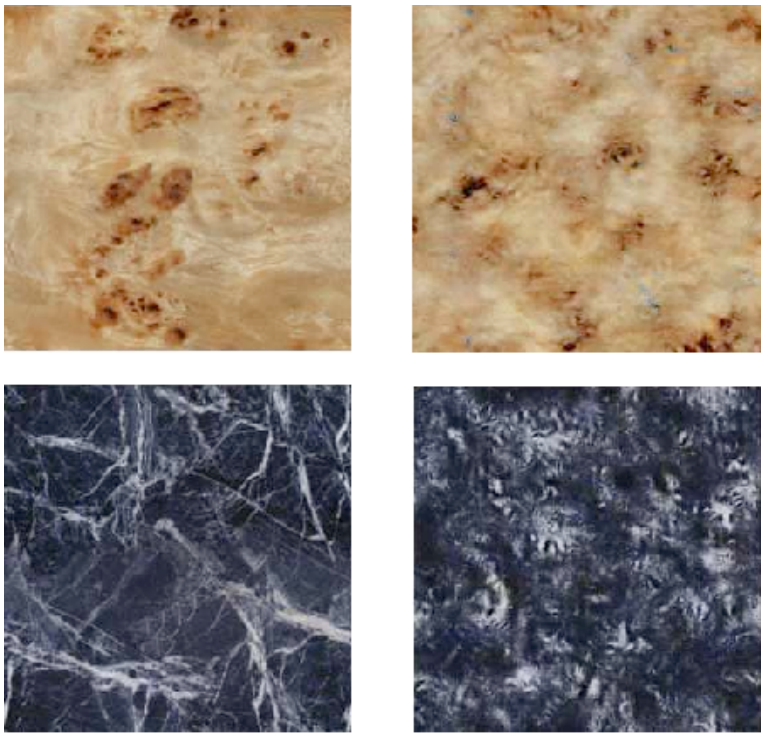


FIGURE 3.2 – Illustration de la méthode paramétrique de synthèse de texture de Heeger et al. ([73]). A gauche, les échantillons sources de texture. A droite, les textures générées.

3.3.2 Les méthodes non paramétriques

Les modèles présentés sont dits non paramétriques dans le sens où la fonction de probabilité des pixels n'est pas explicitement construite : elle est approximée à partir d'un échantillon de texture source. Les textures sont modélisées par un champ de Markov aléatoire, où un pixel est défini exclusivement à partir de ses voisins immédiats (nous supposons que la distribution de probabilité d'un pixel est donnée par la valeur de ses voisins et est indépendante du reste de l'image). Il existe deux grandes familles d'algorithmes de synthèse de texture non paramétriques : les méthodes basées pixel et les méthodes basées bloc.

3.3.2.1 Les méthodes basées pixel

Les méthodes basées pixel ont été initiées par Popat et Picard [113] puis Efros et Leung [57]. Le principe de la méthode décrite dans [57] est le suivant : disposant d'un échantillon de texture, on cherche à synthétiser une nouvelle image pixel par pixel *via* un processus itératif. En supposant un modèle de Markov pour le pixel p à synthétiser, on cherche donc à établir sa distribution connaissant les voisins déjà synthétisés de p, notée $\omega(p)$. La meilleure prédiction de p, notée $\hat{p}$, est définie par la relation :

$$\hat{p} = argmin_{\omega}\ d(\omega(p), \omega),\ \omega \subset I_{ech} \tag{3.1}$$

où I_{ech} est l'échantillon source de texture, ω représente l'ensemble des voisins possibles de I_{ech} et $d(\cdot,\cdot)$ est une fonction de distance entre deux ensembles de pixels (souvent exprimée par une

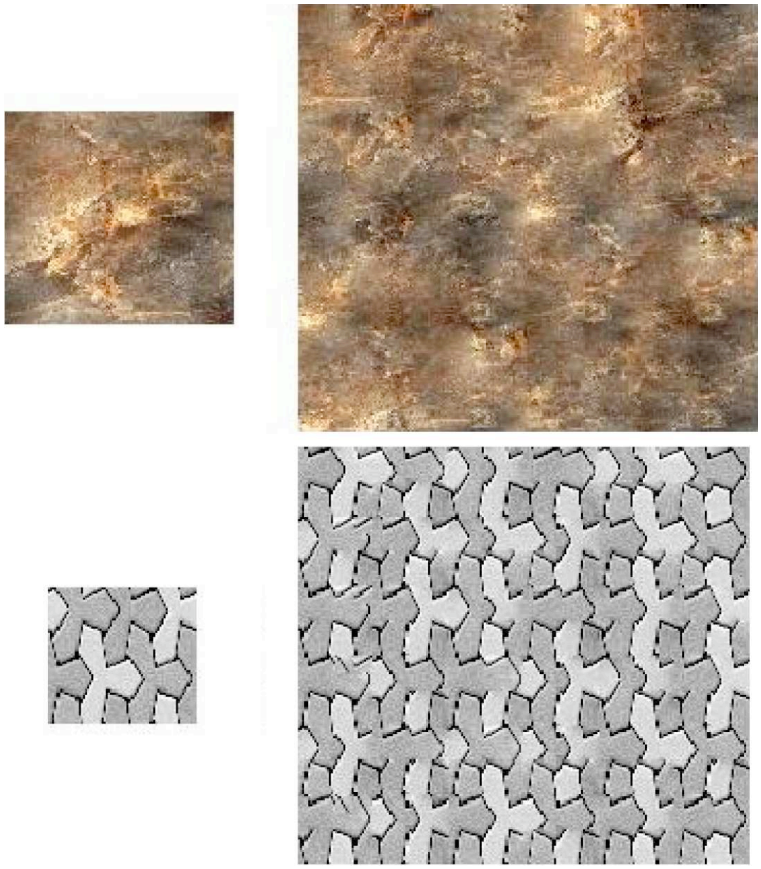


FIGURE 3.3 – Illustration de la méthode paramétrique de synthèse de texture de De Bonet et al. ([29]). A gauche, les échantillons sources de texture. A droite, les textures générées.

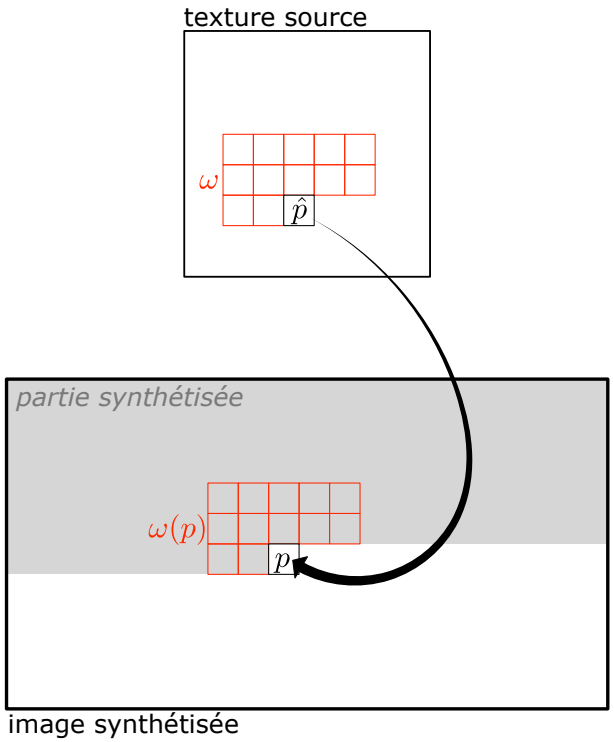


FIGURE 3.4 – Synthèse d'un pixel selon le procédé décrit par Efros et Leung ([57]).

comparaison pixel à pixel de type SAD ou SSD). La valeur du pixel $\hat{p}$ de l'échantillon source est recopiée dans p. Le procédé est itératif jusqu'à ce que tous les pixels de l'image synthétisée soient complétés. La Fig. 3.4 schématise le procédé de synthèse d'un pixel selon un voisinage causal ω prédéfini (la forme en "L", ou *L-shape*). L'unique paramètre de la méthode concerne la taille de la fenêtre du voisinage ω, qui permet d'ajuster le caractère stochastique de la texture synthétisée. Plus le nombre de voisins pris en compte est important et plus l'algorithme est à même de reproduire des textures composées de *texels* de grandes tailles et/ou dispersés, jouant ainsi sur l'aspect rigide ou non de la texture synthétisée. La Fig. 3.5 illustre le résultat obtenu en synthétisant une texture avec plusieurs tailles ω différentes. En augmentant le nombre de pixels voisins pris en compte, la texture générée "semble" plus régulière à cause de la répétition, l'alignement et l'espacement des *texels*. La taille de ω influence directement la complexité de la méthode en terme de nombre de comparaisons nécessaires pour synthétiser un pixel.

Le concept a été étendu par Wei et Levoy [152] en introduisant une approche de synthèse de texture multi-résolution basée sur des pyramides gaussiennes. Le voisinage $\omega(p_i)$ d'un pixel p à un niveau i de la décomposition pyramidale est composé à la fois des pixels de voisinage causal au niveau i (identique à [57]), mais également des pixels du voisinage complet au niveau supérieur $i+1$ de la pyramide. Le procédé itératif de synthèse est de type "*coarse to fine*", c'est-à-dire que les niveaux de la pyramide sont générés du plus haut au plus bas. L'approche proposée combine deux avantages par rapport à la méthode de Efros et Leung [57]. Tout d'abord, la décomposition pyramidale de la texture source "capture" les motifs élémentaires de texture à différents niveaux de résolution, ce qui permet de réduire la taille du voisinage ω tout en garantissant la reproduction de l'aspect régulier des textures déterministes (pour peu que le nombre de niveaux de décomposition de la pyramide soit suffisant). Réduire la taille de ω permet également de réduire le nombre d'opérations nécessaires

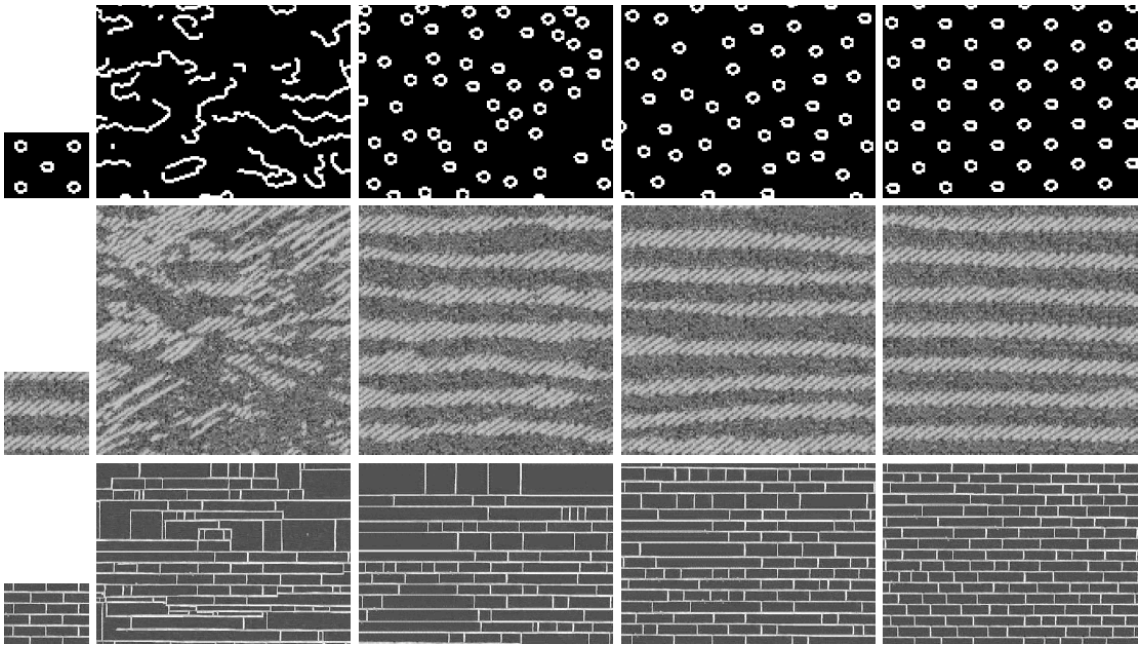


FIGURE 3.5 – Exemples de synthèse de texture selon la méthode proposée par Efros et Leung ([57]). De gauche à droite : échantillon source, image synthétisée avec un nombre de voisins respectivement de 5, 11, 15 et 23 pixels.

à la synthèse d'un pixel, donc la complexité globale de la méthode.

L'algorithme de Wei et Levoy [152] a inspiré Ashikhmin qui propose dans [16] une contrainte supplémentaire permettant de "recopier" des parties entières de l'image source. Au lieu d'effectuer une recherche du meilleur voisinage de p dans l'image source complète, la méthode propose de réduire la recherche aux seuls voisins $\omega(p)$ reliés à une position donnée dans l'image source. Par exemple, considérons qu'un voisinage est composé de quatre pixels répartis en "L" autour du pixel p à synthétiser. A chaque pixel de $\omega(p)$ déjà synthétisé correspond une position dans l'échantillon source. La recherche du plus proche voisinage est restreinte à ces quatre positions dans l'image source, en prenant en compte le décalage de position par rapport au pixel p. La particularité de cet algorithme est qu'il est extrêmement rapide.

Ces méthodes simples s'appliquent à un grand nombre d'applications (synthèse, *warping* 3D, *inpainting*, extension des bords, etc.) et elles donnent généralement de bons résultats visuels. Pour les textures déterministes, la qualité des résultats est influencée par la taille de la fenêtre de recherche qui doit être au minimum aussi grande qu'un motif de texture élémentaire. La complexité en terme de calcul de cette approche basée pixel est telle qu'une utilisation pour une contrainte "temps réel" de synthèse de texture n'est pas envisageable. Pour ces raisons, les méthodes basées bloc sont souvent préférées.

3.3.2.2 Les méthodes basées blocs

Les méthodes paramétriques et les méthodes basées pixel ont montré leurs limites concernant la synthèse de textures déterministes composées de motifs élémentaires irréguliers. La nouvelle approche définie par Efros et Freeman dans [56] veut combler cette lacune en proposant de synthétiser les

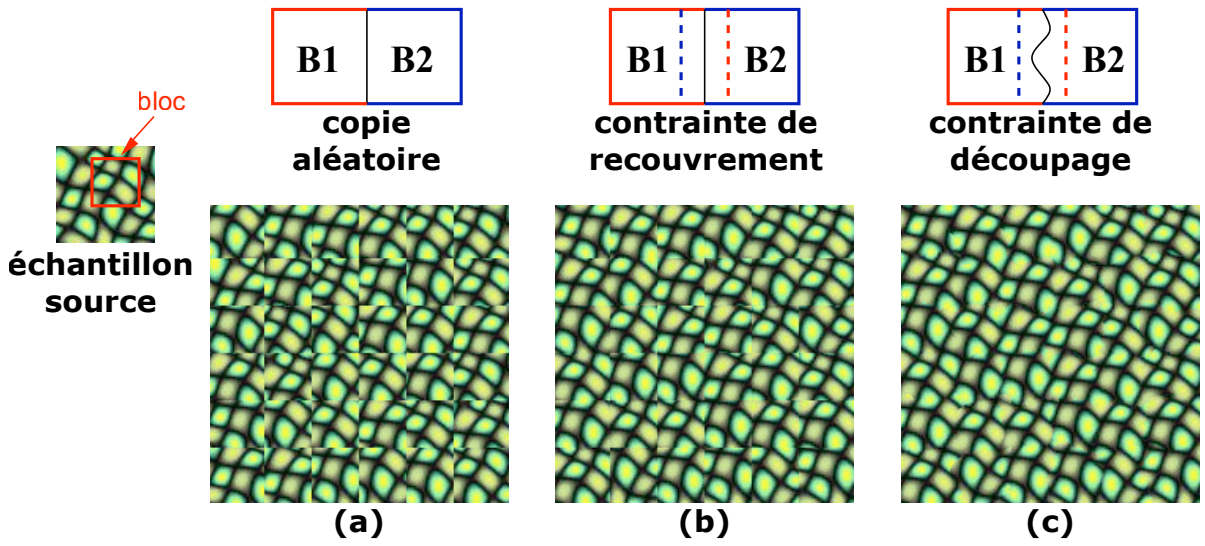


FIGURE 3.6 – Qualité de la texture générée en fonction des contraintes appliquées pour une méthode de synthèse par bloc ([56]).

textures par recopie de bloc (ou *patch*) de pixels, chaque bloc étant constitué d'un ou de plusieurs motifs élémentaires. Pour garantir la continuité des structures de la texture, un bloc à synthétiser dispose d'une zone de recouvrement avec ses blocs voisins. Pour déterminer le bloc de texture source adéquat, une contrainte de minimisation est appliquée sur la zone de recouvrement entre les deux blocs voisins. Cette contrainte est basée sur un critère de différence pixel à pixel, de type SAD ou SSD. Sur la Fig. 3.6(b), nous constatons que la contrainte de minimisation sur la zone de recouvrement ne fait pas totalement disparaître les effets de blocs, bien que ce soit moins flagrant qu'avec une méthode de synthèse sans contrainte (Fig. 3.6(a)). Pour y remédier, une méthode complémentaire consiste ensuite à trouver la meilleure frontière entre les deux blocs à l'intérieur du recouvrement (qui déterminera pour une position donnée son appartenance à un bloc ou l'autre), dans le sens d'une meilleure dissimulation possible des discontinuités. La Fig. 3.6(c) montre que cette technique rend les transitions entre les blocs visuellement indétectables. Finalement, la méthode proposée dans [56] donne de très bons résultats quel que soit le type de texture. L'unique paramètre à définir concerne la taille des blocs utilisée pour la synthèse. Ce paramètre dépend de la taille et de l'espacement des motifs élémentaires de la texture source. Liang et al. ont décrit une méthode similaire dans [85].

Dans [82], Kwatra et al. généralisent le problème de découpage de la zone de recouvrement à l'aide d'un graphe dont les noeuds sont les pixels et les arcs la distance entre pixels voisins (différence entre la "valeur" de deux pixels voisins). De ce fait, un algorithme générique de découpage de graphe permet d'obtenir le meilleur découpage possible, celui qui minimise la distance entre pixels voisins. Par extension, la méthode est utilisée pour synthétiser des vidéos de texture temporelle (c'est-à-dire des textures qui évoluent dans le temps). Dans ce contexte, les blocs sont représentés par des cubes (un même bloc qui évolue dans le temps). La synthèse d'un cube forme une zone de recouvrement spatio-temporelle, qui est représentée par un graphe en trois dimensions. Le procédé a été repris dans [104].

3.3.3 Conclusion

En conclusion, nous avons vu que les méthodes de synthèse de texture peuvent être réparties en deux catégories. Les méthodes paramétriques définissent un jeu de mesures statistiques qui permettent d'analyser un échantillon de texture. La synthèse s'opère ensuite en imposant ces contraintes statistiques sur un bruit blanc. Ce type de méthode offre l'avantage de pouvoir restreindre les caractéristiques d'une texture à un nombre limité de paramètres, mais peine à synthétiser des textures composées de motifs.

L'autre grande catégorie de méthodes est basée sur la corrélation qui unit un *patch* (un pixel ou un bloc) à ses voisins. Pour synthétiser un *patch*, on recherche dans l'échantillon source le voisinage causal le plus proche, puis une recopie est effectuée entre le *patch* associé de l'image source et le *patch* à synthétiser.

Dans les deux types d'approches, la décomposition multi-échelle du signal d'entrée permet d'analyser et de synthétiser une texture à différents niveaux de résolutions. De meilleurs résultats visuels sont généralement constatés.

Le papier [87] décrit différentes méthodes de synthèse de texture et propose de comparer visuellement les résultats de quelques unes d'entre elles.

3.4 La restauration par synthèse d'image

3.4.1 Etat de l'art

L'objectif d'une méthode de restauration est d'obtenir un algorithme automatique permettant de "remplacer" certaines régions d'image définies par l'utilisateur, par exemple supprimer un défaut dans l'image, supprimer un objet non désiré, etc. L'efficacité de l'algorithme est mesurée par sa capacité à "remplir" la région ciblée d'une manière visuellement indétectable et naturelle. Cependant, le constat dressé montre que les articles présents dans la littérature se focalisent sur un des deux aspects de l'inpainting ou de la synthèse :

- La synthèse de texture permet de générer des images à l'infini visuellement identiques. Cependant, ces algorithmes montrent leurs limites lorsqu'ils sont appliqués sur des images naturelles, composées de textures mixtes et non "pures".
- L'inpainting permet de combler les trous d'images en propageant les structures (isophotes). L'inconvénient est que ces algorithmes, inspirés des équations aux dérivées partielles, introduisent du flou, notamment lorsqu'il s'agit de larges portions d'image à restaurer (les méthodes les plus connues sont présentées dans [23, 22, 38, 143]).

La famille de méthodes auxquelles nous allons nous intéresser s'inspire directement de méthodes non paramétriques de synthèse de texture et arrive à palier les problèmes évoqués ci-dessus. La partie visible de l'image sert de base d'apprentissage pour en déduire la partie manquante. Le traitement est itératif jusqu'à ce que l'image soit intégralement restaurée.

Bien qu'inspirée du domaine de la synthèse, cette famille de méthodes doit être capable de prendre en compte des images naturelles bien plus complexes que des textures. Cette catégorie d'applications est aussi appelée "synthèse contrainte". L'un des premiers travaux portant sur ce domaine fut proposé par Harrison dans [71]. Inspiré de l'algorithme de synthèse de texture basé pixel de Efros et Leung [57], il introduit l'idée que, dans ce contexte d'utilisation, l'ordre de synthèse des pixels doit être contraint

par la relation qui les unit à leurs voisins. Pour cela, une analyse de l'image disponible permet d'extraire les redondances statistiques entre pixels voisins. Les redondances les plus présentes sont favorisées lors du processus de restauration. Ce mode opératoire représente une rupture par rapport aux algorithmes itératifs de synthèse classiques soumis à un ordre de traitement fixe (*raster scan*). Le concept d'attribuer une priorité de restauration en fonction de critères intrinsèques à la géométrie de l'image va être repris dans les méthodes suivantes. L'article [31] propose un modèle plus simple où la priorité de chaque pixel est définie par le nombre de voisins directs connus. Cette contrainte garantit un ordre de restauration de l'extérieur vers l'intérieur de la zone à remplir, assurant une cohérence visuelle dans le processus de reconstruction.

Une méthode multi-résolution a été proposée dans l'article de Drori et al. [54]. Une image basse résolution est d'abord complétée, puis sert d'approximation pour l'image au niveau de résolution supérieur. Pour chaque niveau de résolution, deux opérations successives sont réalisées :

1. L'information connue, aux contours des zones à restaurer, est interpolée pour obtenir une approximation des parties manquantes. La méthode employée est un procédé de filtrage itératif, adapté au niveau de résolution, présenté dans [68], et qui donne un rendu lisse des régions complétées.
2. Une méthode de synthèse basée *patch* est appliquée afin d'ajouter des détails aux régions manquantes préalablement lissées. Les *patchs*, de forme circulaire, sont composés de pixels connus et de pixels à synthétiser. Le fait d'avoir préalablement approximé les pixels à synthétiser permet d'affiner la recherche du meilleur *patch* source.

Un concept intéressant développé dans la méthode de [54] concerne la taille des *patchs* utilisés, qui s'adapte en fonction du contenu local de l'image (plus récemment traité dans [165]). L'objectif est d'obtenir une taille de *patch* inversement proportionnelle à la fréquence spatiale de la portion d'image en question : les régions "plates" se voient donc attribuer des *patchs* de plus grande taille. Pour mesurer l'activité locale de l'image, une méthode très simple consiste à calculer la différence entre les valeurs extremums des pixels du *patch*.

Pour remédier à la complexité de l'algorithme présenté dans [54], Criminisi et al. ont proposé un nouvel algorithme dans [50]. Les auteurs évoquent deux paradigmes qui posent les bases de leur méthode. Tout d'abord, la diffusion des isophotes de l'extérieur vers l'intérieur de la zone à restaurer est un élément essentiel garantissant un résultat visuellement cohérent. Contrairement aux algorithmes qui "diffusent" les contours pixel à pixel, les auteurs se sont rendus compte que la synthèse de texture basée modèle par recopie de groupes de pixel était suffisante pour assurer la propagation des structures. Ensuite, le concept important que les auteurs ont mis en évidence est que l'ordre de reconstruction des pixels dans un schéma de restauration d'image est critique. Alors que la littérature est constituée d'algorithmes qui prennent peu en compte cet aspect (le plus souvent en recourant à la méthode de "pelure d'oignon", c'est-à-dire en procédant par couches successives de l'extérieur vers l'intérieur de la zone à restaurer), l'article présente une méthode plus évoluée qui va privilégier la reconstruction des pixels (a) dont on connaît le voisinage et (b) qui se situent sur un contour. Ainsi, la diffusion des structures est assurée en premier, et l'information du voisinage garantit une synthèse efficace. Le juste équilibre entre les critères (a) et (b) est la clé de cette approche. La description de l'algorithme est présentée ci-dessous.

Pour reprendre les notations utilisées dans le domaine de l'inpainting, la région à restaurer est notée Ω et son contour $\delta\Omega$. La région restante, dite "source" et notée Φ, fournit les échantillons nécessaires pour la synthèse. Enfin, nous notons Ψp le *patch* centré sur le pixel p. La taille de la fenêtre Ψp doit être spécifiée pour contenir au moins un élément de texture ("texel") dans la région source. Dans la

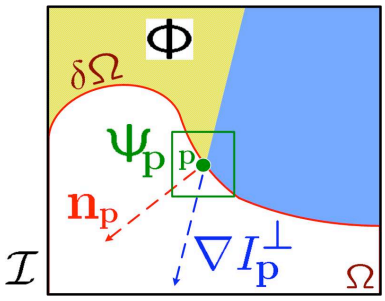


FIGURE 3.7 – Calcul de la priorité de restauration pour chaque pixel p, où n_p est le vecteur normal au contour $\delta\Omega$ et $\nabla I_p^{\perp}$représente l'isophote au point p.

majorité des cas, une taille de fenêtre 9×9 est un bon compromis entre résultats et performances. A chaque pixel p à restaurer est attribuée une priorité définie par :

$$P(p) = C(p)D(p) \tag{3.2}$$

où $C(p)$ est appelé le critère de confiance et $D(p)$ le critère de données, qui permet de privilégier la reconstruction des pixels le long des contours. Le critère de confiance $C(p)$ privilégie les *patchs* Ψp qui disposent de plus de pixels connus (notés $q \in \Psi p \cap \Phi$). En effet, chaque pixel connu utilisé pour la synthèse est un critère discriminant supplémentaire. Leur nombre influence directement la qualité de la reconstruction. $C(p)$ est défini comme suit :

$$C(p) = \frac{\sum_{q \in \Psi_p \cap \Phi} C(q)}{|\Psi_p|} \tag{3.3}$$

où $|\Psi_p|$ est l'aire de Ψ_p.

Le critère de données $D(p)$ est donné par :

$$D(p) = \frac{|\nabla I_p^{\perp} \cdot n_p|}{\alpha} \tag{3.4}$$

où α est un facteur de normalisation (par exemple $\alpha = 255$ pour une image en niveaux de gris). n_p est un vecteur unitaire orthogonal à $\delta\Omega$ au point p et $\perp$ est l'opérateur d'orthogonalité. $\nabla I_p^{\perp}$ représente un vecteur dont la direction et la valeur sont définies par l'orientation et l'intensité de l'isophote au point p. La figure 3.7 illustre n_p et $\nabla I_p^{\perp}$. Pour l'initialisation, la fonction $C(p)$ est définie par $C(p) = 0,\ \forall p \in \Omega$ et $C(p) = 1,\ \forall p \in \Phi$. Le critère $C(p)$ garantit un ordre de restauration concentrique et le critère $D(p)$ favorise la diffusion des isophotes orthogonaux à la frontière $\delta\Omega$ entre la partie visible et la partie à restaurer. Pour chaque itération de l'algorithme, la première étape consiste à identifier le pixel $\hat{p}$ avec la priorité $P(\hat{p})$ la plus élevée.

L'algorithme se charge ensuite de compléter la partie manquante de $\Psi_{\hat{p}}$ avec des données extraites de Φ. Pour cela, nous recherchons dans la région source le *patch* candidat le plus similaire à $\Psi_{\hat{p}}$, c'est-à-dire :

$$\Psi_{\hat{q}} = \arg \min_{\Psi_q \in \Phi} d(\Psi_{\hat{p}}, \Psi_q) \tag{3.5}$$

Algorithme 3.1 Méthode d'*inpainting* proposée dans [50].

1. Identifier la frontière $\delta\Omega^t$. Si $\delta\Omega^t = \varnothing$, terminer l'algorithme.
2. Calculer les priorités $P(p)\ \forall p \in \delta\Omega^t$.
3. Trouver la fenêtre $\Psi_{\hat{p}}$ avec la priorité maximale.
4. Trouver le modèle source $\Psi_{\hat{q}} \in \Phi$ qui minimise $d(\Psi_{\hat{p}}, \Psi_{\hat{q}})$.
5. Copier les pixels de $\Psi_{\hat{q}}$ vers $\Psi_{\hat{p}} \forall p \in \Psi_{\hat{p}} \cap \Omega$.
6. Mettre à jour $C(p) \forall p \in \Psi_{\hat{p}} \cap \Omega$.

FIGURE 3.8 – Comparatif de deux méthodes de suppression de larges régions dans une image. De gauche à droite : image d'origine, masque désignant la portion à supprimer, résultat de la méthode de Criminisi et al. ([50]), résultat de la méthode de Bertalmio et al. ([23]).

où la distance $d(a, b)$ entre deux fenêtres a et b est la somme des différences au carré (SSD) entre les pixels connus des *patchs*. Une fois la source $\Psi_{\hat{q}}$ trouvée, les valeurs des pixels de $\Psi_{\hat{p}}$ à compléter, tel que $p'|p' \in \Psi_{\hat{p}} \cap \Omega$, sont copiées à partir de leur position correspondante dans $\Psi_{\hat{q}}$.

Pour résumer, l'algorithme complet est explicité dans 3.1.

L'algorithme présenté par Criminisi et al. est intéressant sur plusieurs points. Les efforts se sont focalisés en premier lieu sur l'ordre de restauration des pixels plus que sur la méthode de synthèse en elle-même. Les résultats montrent en effet que leurs hypothèses sont bonnes : la méthode permet à la fois de synthétiser les textures et de restaurer les contours. Au final, l'algorithme est capable de restaurer de larges portions d'image naturelle composées de textures mixtes. Le principal reproche que nous pouvons évoquer est l'impossibilité de restaurer correctement des lignes de contours incurvées, à cause de l'utilisation de la méthode de synthèse par *patch*, et à l'inverse de certains autres procédés ([78, 39]). Des résultats de la méthode sont montrés sur la Fig. 3.8. Au final, l'image reconstruite semble plus "naturelle" qu'avec les méthodes de diffusion basées sur les équations aux dérivées partielles.

3.4.2 Contribution : adaptation des critères d'ordre dans une méthode de restauration non paramétrique

Dans la méthode de restauration présentée par Criminisi et al. [50], et qui à ce jour présente l'une des meilleurs solutions de restauration de larges régions dans une image, nous avons vu que la principale innovation reposait sur l'ordre de restauration des *patchs*. Pour cela, deux critères sont proposés (voir Eq.3.3 et Eq.3.4), l'un permettant de reconstruire la partie manquante de l'extérieur vers l'intérieur (*confidence term*), l'autre permettant de propager les isophotes en priorité (*data term*). Nous avons modifié ces deux critères afin d'améliorer cette méthode.

Dans la méthode [50], le gradient ∇I_p du *patch* courant, qui permet de mesurer l'intensité et l'orientation du ou des isophote(s), est calculé comme la valeur maximale du gradient dans $\Psi_p \cap \Phi$. Dans ce cas, nous constatons que ∇I_p ne différencie pas les textures aléatoires des structures de l'image, particulièrement quand la texture est composée de contours fortement marqués. Pire, ∇I_p ne reflète pas forcément la direction "globale " de $\Psi_p \cap \Phi$. Dans cette contribution, nous proposons d'utiliser plutôt :

$$\vec{s} = \sum_{p' \in \Psi_p \cap \Phi} \nabla I_p \tag{3.6}$$

Pour un contour orienté, la valeur de $\vec{s}$ sera importante, donc le *patch* Ψ_p aura une priorité plus importante. A l'inverse, pour une texture marquée avec des contours dans toutes les directions, la valeur de $\vec{s}$ sera petite.

La deuxième amélioration de la méthode de Criminisi et al. concerne la configuration géométrique des *patchs,* que nous avons choisie circulaire contrairement aux *patchs* carrés de l'article de référence. Ce nouveau choix est motivé par deux observations :

- L'algorithme privilégie la reconstruction de fenêtres dont un maximum de voisins est connu (voir l'Eq. 3.3). Or, comme il est illustré sur la figure 3.9, le format de type carré a tendance à favoriser les *patchs* adjacents par rapport aux *patchs* déjà reconstruits. En effet, sur la sous-figure (2) utilisant des *patchs* carrés, le terme de confiance C est maximum pour le point c. Cet effet est minimisé par l'utilisation de *patchs* circulaires (sous-figure (2)). Au final, l'ordre de priorité va être le même dans les deux cas, c'est-à-dire $C(a) < C(c) < C(b)$ et $C(a') < C(c) < C(b)$, mais leurs valeurs seront plus nuancées dans le cas des *patchs* circulaires. Ainsi, le critère de mesure des contours aura plus de poids pour ordonner les *patchs*.
- Dans certains cas, l'utilisation de *patchs* carrés peut entraîner des effets de blocs visuellement gênants. Ces effets de blocs sont atténués *via* l'utilisation de *patchs* circulaires plus adaptés aux structures d'images naturelles.

La figure 3.10 illustre l'amélioration de la méthode de Criminisi et al. par notre contribution. La taille des *patchs* Ψ_p est fixée à 9×9 pixels. D'autres illustrations sont présentées en annexe de ce manuscrit.

3.4.3 Conclusion

Les algorithmes tels que celui développé dans [50] présentent un compromis efficacité/complexité raisonnable. Il est intéressant de noter qu'à ce jour, ces recherches ont donné lieu à des produits commerciaux permettant d'améliorer une image par suppression d'objets. Par exemple, la fonctionnalité nommée *patchmatch* dans le produit Photoshop de la société Adobe a constitué un argument

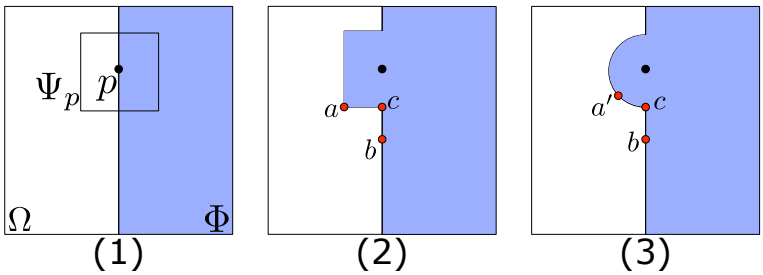


FIGURE 3.9 – Impact de la configuration géométrique des *patchs* sur le calcul de l'indice de confiance de la méthode de Criminisi et al. ([50]).

marketing pendant la sortie de la version CS5 en avril 2010. D'autres références, telles que InPaint ou iPhoto, utilisent également ce nouveau genre d'outils pour la retouche d'image.

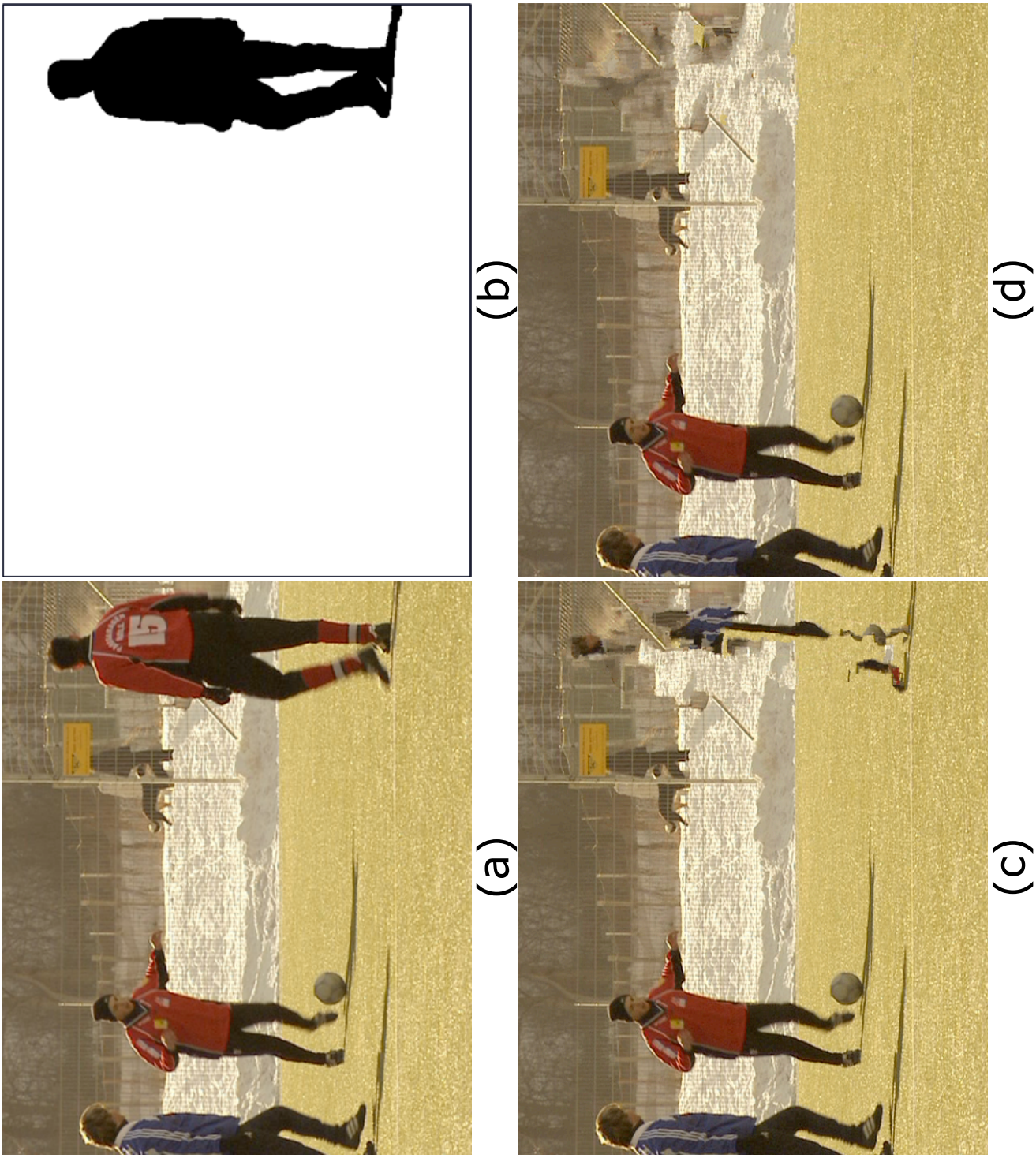


FIGURE 3.10 – Illustration de l'amélioration de la méthode de Criminisi et al. ([50]). (a) image d'origine (352 × 288 pixels) (b) masque (en noir la partie à restaurer) (c) méthode de Criminisi et al. (d) notre contribution.

3.5 Codage vidéo par analyse/synthèse de contenu

Comme nous avons pu le constater dans les sections précédentes, il existe une multitude de procédés de synthèse et de restauration d'image plus ou moins adaptés au type de contenu que nous souhaitons traiter (texture aléatoire/rigide, image naturelle ou séquence d'images). Par la suite, ces méthodes ont donné lieu à des recherches spécifiques portant sur le domaine de la compression d'image, et par extension de vidéo. Le concept général repris par toutes ces recherches consiste à supprimer volontairement certaines portions d'image à l'encodeur, économisant ainsi le coût de codage de l'information texturelle. Ces portions sont restituées au décodeur grâce au recours à des méthodes de synthèse et d'inpainting. Nous allons voir qu'il existe une multitude de façons d'intégrer ces méthodes dans un schéma de compression d'image, selon le degré d'imbrication de la méthode de restauration au sein de l'encodeur. Nous allons balayer les méthodes les plus couramment citées dans la littérature, en prenant soin d'analyser leurs limites. Cette réflexion nous mènera à identifier un axe de recherche prometteur, qui a donné lieu au travail de contribution proposé dans le chapitre suivant.

Les approches que nous allons énumérer sont réparties en deux familles. La première propose des méthodes de codage en rupture avec les schémas de codage existants. Elles reposent en général sur une pré-analyse fine du contenu en entrée, qui se traduit par une segmentation spatiale de l'image. Chaque région extraite est soit codée par une méthode classique, soit synthétisée. Ainsi, selon la définition même de la synthèse de texture, l'efficacité de la méthode est mesurée uniquement selon des critères purement visuels. La seconde famille propose une série de méthodes plus souples, naturellement intégrées au sein des schémas de codage conventionnels.

3.5.1 Les schémas de codage basés contenu avec évaluation perceptuelle

Le postulat sous-jacent de ce type de solutions admet qu'une texture qui compose une image peut être remplacée par une autre qui lui "ressemble". Par exemple, l'herbe d'un terrain de foot ne doit pas nécessairement être identique à l'originale pour que le téléspectateur l'interprète comme telle. En fait, n'importe quelle texture d'herbe (plus claire ou plus foncée, présentant des motifs différents...) suffira à duper le système visuel humain pour que la scène semble naturelle. Avec ce type de formulation, nous retombons dans l'expression du problème de synthèse de texture. L'intérêt que nous avons à l'utiliser dans un contexte de codage d'image réside dans le fait qu'une texture peut être représentée par un nombre restreint d'information (comme nous l'avons vu précédemment), et peut donc contribuer à réduire le coût de codage d'une image.

Ces méthodes ne donnent pas à elles seules de résultats de codage satisfaisants : elles sont la plupart du temps couplées à des méthodes de codage classiques (codage hybride) avec lesquelles elles entrent en coopération ou en compétition. Un schéma de codage vidéo basé objet, de type MPEG-4 *Visual* (présenté et illustré dans la section 2.3.3), est particulièrement bien adapté à la représentation et à la compression de ce type de codage hybride. Pour tirer bénéfice de ces méthodes, de nouvelles techniques d'évaluation de la qualité sont par ailleurs nécessaires (voir [102]) car l'erreur quadratique moyenne, utilisée classiquement en codage, n'est plus pertinente pour des contenus synthétisés ou mixtes.

Nous pouvons faire le lien entre ces méthodes et les algorithmes de dissimulation d'erreur de transmission. L'objectif de remplir de façon naturelle des régions manquantes d'une image est commun,

mais le type d'application est différent. Dans le cas qui nous intéresse, la finalité est d'améliorer les performances en terme de compression des encodeurs actuels.

Les méthodes présentées ci-dessous reposent sur une analyse sémantique de la scène, en prenant en compte les mouvements de caméra et/ou le déplacement des objets de la scène. Cette étape est nécessaire car elle assure une cohérence temporelle des éléments synthétisés : une texture créée dans une frame t doit être recopiée à l'identique dans toutes les frames suivantes au risque de créer un effet de scintillement à la lecture de la séquence. Ceci implique une segmentation spatio-temporelle assez lourde et non robuste, ce qui est à notre sens une des limites de ce type d'approche.

Dumitras et Haskell [55] ont proposé une méthode de compression vidéo par remplacement de texture. En amont de l'encodeur, une analyse permet d'identifier et d'extraire les textures de l'image. La séquence résultante est codée de façon classique, et les paramètres statistiques des textures supprimées sont transmis. Au décodeur, les régions manquantes sont synthétisées par une approche paramétrique et insérées dans la séquence. Bien que des gains significatifs soient obtenus pour certaines configurations d'encodage et types de contenus (jusqu'à 55% par rapport à H.264/AVC), les contraintes sur la taille des textures supprimées (au minimum 40% de la taille des images pour être rentable, sur au moins 50 images), le peu de fiabilité de la méthode de segmentation spatio-temporelle des textures et la nécessité d'avoir une séquence sans mouvement global font que la méthode n'est pas viable pour un codeur vidéo générique. De plus, aucun contrôle sur la qualité visuelle de la texture ne permet de garantir à tous les coups que la méthode de synthèse soit efficace.

Zhu et al. [166, 167] ont proposé un schéma de codage vidéo basé sur H.264/AVC et intégrant une méthode de détection et de synthèse non-paramétrique de texture sur les frames de type "B", selon un partitionnement par bloc 8×8 pixels. Les frames "I" et "P" d'un GOP [2] sont utilisées en tant qu'échantillon source pour la synthèse de texture. En exploitant les informations de compensation en mouvement des blocs, chaque GOP subit une segmentation spatio-temporelle reprise de [91]. Ainsi, une représentation temporelle adéquate permet de suivre une même portion d'image tout au long du GOP, indépendamment des mouvements de caméra et d'objets dans la séquence. Chaque bloc est classifié en tant que "structure" ou "texture" en utilisant un algorithme simple de détection de contour. Les blocs de texture seront synthétisés, alors que les structures seront codées par H.264/AVC, au même titre que les frames "I" et "P". Cette étape de segmentation est la clé principale de ce type de méthode car elle assure la stabilité spatiale et temporelle des textures synthétisées de la séquence. En effet, le procédé de synthèse prend en compte les aspects aléatoires d'une texture, ce qui a pour conséquence de ne pas pouvoir reproduire exactement une texture (au sens du PSNR) dans deux frames différentes. Dans ce cas, cela se traduit par une gêne perceptuelle qui donne l'impression qu'une texture "bouge" dans le temps. Ce point est un handicap par rapport aux même types de méthodes cantonnées à la compression d'images fixes [89, 90, 151], beaucoup plus efficaces pour ce type d'application. Basée sur une estimation visuelle de la qualité, la méthode proposée par Zhu et al. peut atteindre 38.8% de gain de débit à qualité équivalente par rapport à H.264/AVC.

La segmentation spatio-temporelle basée bloc est reprise dans [101] (qui donne suite aux travaux des mêmes auteurs dans [102, 103, 100, 104]). De la même façon que dans Zhu et al., les frames "I" et "P" sont codées en respectant le standard H.264/AVC, selon le principe débit-distorsion basé sur une mesure MSE. Seules les frame "B" sont traitées dans la boucle de codage par analyse-synthèse. Après analyse d'un GOP de la séquence, la méthode propose deux types de synthèses pour les textures respectivement rigides et non rigides dans le temps (e.g. l'eau, le feu, ...) ; la synthèse de texture rigide est notée ST_r, et de façon complémentaire ST_{nr}).

2. *Group Of Pictures*

En ce qui concerne la synthèse de texture rigide ST_r, l'analyse du GOP a permis d'établir un masque de segmentation pour chaque frame. Ensuite, un traitement reposant sur un procédé de *warping* (ou déformation), basé sur un modèle de mouvement présenté dans [106], permet de faire correspondre une région de texture d'une frame de référence à une région de texture d'une frame à synthétiser. Les parties manquantes de texture, à cause d'effets de recouvrement-découvrement d'objets, sont synthétisées en utilisant une méthode non-paramétrique basée pixel (du type 3.3.2.1). Les informations transmises pour permettre de reconstruire l'image au décodeur sont, par type de texture, le masque de segmentation correspondant, les paramètres de mouvement (quantifiées et déquantifiées), et une référence pour indiquer la frame source.

Le module ST_{nr} est inspiré des travaux de Kwatra et al. [82] (voir 3.3.2.2) pour permettre de synthétiser un volume $2D + t$ de texture. Dans ce contexte, les images disponibles à l'extrémité du GOP définissent un état de départ et d'arrivée de la texture pour contraindre le procédé de synthèse, ainsi qu'un ensemble d'échantillons sources de texture. Les informations nécessaires au décodeur pour synthétiser chaque texture non rigide sont le masque de segmentation et ses paramètres de mouvement pour assurer l'alignement temporel de la texture.

Pour estimer la qualité des textures synthétisées, Ndjiki et son équipe ont mis au point un outil de contrôle de qualité visuelle VQA [3] basé sur un modèle psychovisuel inspiré de [109]. La méthode, présentée comme à peine trois fois plus complexe qu'une mesure de PSNR, permet de mettre en évidence les artefacts introduits par le procédé de synthèse. Grâce à cet outil, la qualité des deux procédés de synthèse ST_r et ST_{nr} peut être mesurée. En cas d'échec de synthèse selon un critère débit-distorsion, l'encodeur conventionnel H.264/AVC est employé pour transmettre la texture.

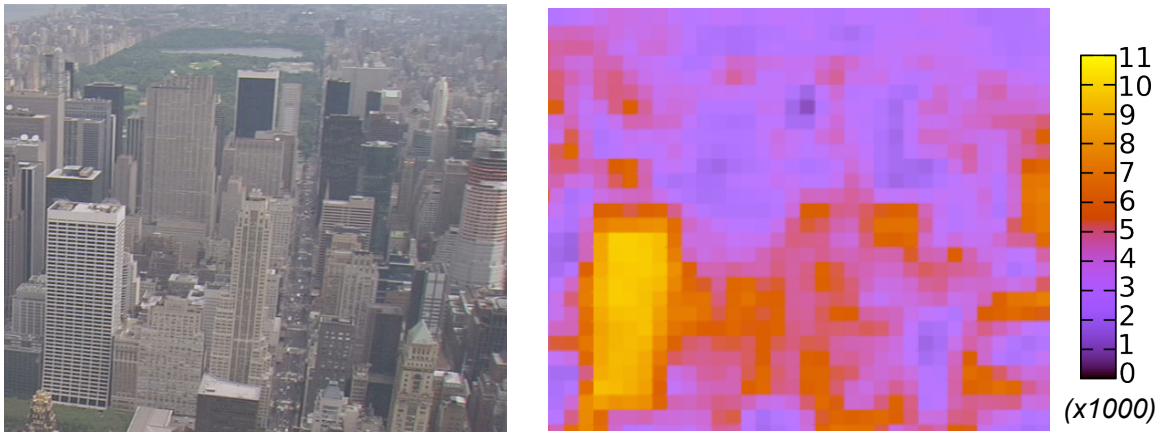
A notre sens, plusieurs difficultés montrent que ce type d'approche possède ses limites dans un contexte de codage vidéo :

– difficulté à proposer une analyse sémantique de la scène robuste et efficace.
– difficulté à proposer une métrique de qualité prenant en compte des aspects psychovisuels.
– schémas de codage hybrides qui posent des problèmes d'intégrations avec H.264/AVC.

Sur ce dernier point, nous avons vu que le mécanisme interne du standard H.264/AVC est en grande partie basé sur la prédiction (2.3.2) en prenant en compte les blocs voisins préalablement codés (que ce soit en intra ou en inter). Or, en proposant une alternative au codage, ce mécanisme de prédiction est rompu et affaiblit les performances en compression de la partie H.264/AVC. Enfin, l'assemblage de régions codées par H.264/AVC et de textures synthétisées au sein d'une séquence peut engendrer une gêne perceptuelle en formant un ensemble peu naturel. Par exemple, nous pouvons concevoir qu'une texture très détaillée qui est synthétisée dans un contexte de codage à bas débit puisse contraster avec la portion très dégradée de la séquence. Il s'agit ici de définir un juste équilibre du compromis débit-distorsion entre les deux approches de codage, alors qu'aucune métrique de qualité viable, pour mesurer une texture synthétisée, ne semble faire l'unanimité.

En conclusion, bien que les méthodes décrites ci-dessus soient élégantes dans leur approche et qu'elles proposent un schéma de codage très innovant, il est difficile avec de tels mécanismes de proposer un encodeur générique satisfaisant. Il y a, à notre sens, beaucoup de verrous techniques qui en limitent les résultats : mesure de qualité subjective, segmentation spatio-temporelle robuste, etc. La section suivante tente de remédier à ces contraintes avec une approche différente.

3. *Video Quality Assessor*



(a) Portion de l'image d'origine (b) Coût débit-distorsion

FIGURE 3.11 – Illustration du coût débit-distorsion par macro-bloc de H.264/AVC en codage intra-image.

3.5.2 Les schémas de codage basés sur une qualité objective

Dans cette section, nous allons évoquer un schéma de codage inspiré des algorithmes de synthèse et de restauration d'image. La différence avec la section précédente peut être décrite par le niveau d'imbrication plus élevé au sein du codeur-décodeur H.264/AVC. Alors que l'approche précédente consistait à proposer une alternative au codage des textures, avec son mécanisme propre de segmentation et de mesure de qualité, les méthodes décrites ici proposent un outil prédictif supplémentaire aux prédictions intra et inter classiques. Ceci étant, la mesure de la qualité de la prédiction objective (de type MSE), l'estimation du coup de codage et la fonction d'optimisation *RD-opt* sont repris du standard vidéo H.264/AVC. L'intérêt d'utiliser ces mécanismes bien connus est de se détacher des problèmes de mesure de qualité, basés sur des aspects subjectifs, et de la segmentation spatio-temporelle complexe et peu robuste. Le but est d'apporter de nouvelles méthodes de prédiction qui se prêtent bien à l'estimation de textures rigides composées de motifs répétés, là où H.264/AVC, par son système de prédiction intra *via* une interpolation des pixels voisins, montre ses limites. Pour illustrer ces propos, la figure 3.11 montre le coût débit-distorsion par macro-bloc d'un extrait de la séquence *City* (en 720*p*) dans un contexte de codage uniquement intra-image. Le coût débit-distorsion optimal est obtenu grâce à la fonction *RD-opt* [134], notée J, s'exprimant par :

$$\min J, \text{ où } J = D + \lambda R, \tag{3.7}$$

avec D représente la distorsion du macro-bloc, exprimée par une comparaison pixel à pixel de type SAD, R le débit exprimé en bits et λ un paramètre de Lagrange. Sur la figure 3.11, nous constatons que la texture rigide au premier plan, représentant l'immeuble en bas à gauche de l'image, possède un coût débit-distorsion très élevé, d'un facteur deux à trois par rapport au reste de l'image. Cet exemple illustre l'incapacité des méthodes de prédiction intra-image de H.264/AVC à modéliser les textures rigides.

Les méthodes décrites dans cette section, appelées prédiction par *template matching* ou par modèle markovien de texture, répondent à ce besoin. Au niveau de leur mise en oeuvre, elles entrent en

compétition avec les méthodes de prédiction H.264/AVC. Dans un premier temps, nous nous focaliserons sur les méthodes de prédiction intra par *template matching*, avant d'étendre le concept à la prédiction temporelle.

3.5.2.1 Principe du *template matching*

La prédiction par *template matching* s'inspire des travaux sur la synthèse de texture non paramétrique basée *patch,* introduits dans la section 3.3.2.2, qui reposer sur le formalisme des champs de Markov aléatoires. L'hypothèse donnée est qu'une portion de texture p (un ensemble de pixels) peut être caractérisée par son voisinage $T(p)$ appelé *template*. Dans l'échantillon de texture source dont nous disposons, il existe un *template* $T(\hat{q})$, aussi appelé *L-shape,* tel que :

$$T(\hat{q}) = \arg \min_{T(q) \in S} d(T(p), T(q)) \tag{3.8}$$

où la distance $d(T(a), T(b))$ entre deux ensembles $T(a)$ et $T(b)$ est définie par une comparaison des valeurs de pixel à la même position et S représente l'échantillon source de texture. La méthode de recherche de $T(\hat{q})$ revient à chercher le *template* source le plus proche de $T(p)$ dans l'échantillon de texture. Parce que $T(\hat{q})$ caractérise $\hat{q}$ et que $T(p)$ et $T(\hat{q})$ sont proches, alors nous supposons que $\hat{q} \approx p$, et le procédé repose sur une simple recopie des pixels de $\hat{q}$ vers p.

Sur ce principe, Tan et al. [137] (et simultanément [158]) ont élaboré un algorithme de prédiction spatiale qu'ils ont intégré dans un codeur vidéo compatible H.264/AVC. Un nouveau mode de prédiction par *template matching* pour les blocs de luma de taille 4×4 et 8×8 pixels est ajouté à la liste des neufs modes de prédiction intra de H.264/AVC (voir 2.3.2.2). Pour cela, chaque bloc 4×4 et 8×8 est partitionné en sous-blocs 2×2 prédits à partir de leurs cinq pixels adjacents disponibles (Fig. 3.12). L'image source S est tout ou partie de l'image en cours, partiellement codée/décodée (voisinage causal selon le principe de parcours des blocs dans l'image, appelé *raster scan mode*). Ainsi, le procédé de recherche du plus proche template $T(\hat{q})$ est réalisé de façon identique au codeur et au décodeur afin d'obtenir le même prédicteur $\hat{q}$. La seule information à transmettre au décodeur concerne la signalisation du mode de prédiction intra par *template matching*, ce qui se traduit par le codage d'un *flag* de signalisation supplémentaire par bloc 4×4 et 8×8 lorsque la prédiction intra est utilisée.

Concernant la complexité de calcul de la méthode de prédiction par *template matching*, la méthode à l'encodeur est similaire à l'estimation de mouvement de H.264/AVC en terme de nombre d'opérations (la recherche de blocs proches en inter ou de *template* proches en intra est finalement comparable). En revanche, au décodeur, la même opération de recherche du plus proche *template* doit être effectuée. L'augmentation de la complexité est donc dépendante du pourcentage d'utilisation du mode de prédiction par *template matching*, qui lui même dépend de la séquence codée. En effet, la méthode proposée est plus adaptée à prédire certains types de textures rigides, dont la présence varie d'une séquence naturelle à l'autre (le pourcentage d'utilisation de la méthode varie entre 20 et 40%, ce qui reste tout de même beaucoup plus élevé que les autres modes de prédiction intra-image). La complexité en calcul est aussi dépendante de la taille de la fenêtre de recherche S.

Dans un contexte de codage vidéo H.264/AVC FRext (CABAC et DCT 8×8 activés) où toutes les frames sont de type "I"), les gains en débit à qualité équivalente peuvent atteindre 11% [137]. Une implémentation très proche est proposée dans [158].

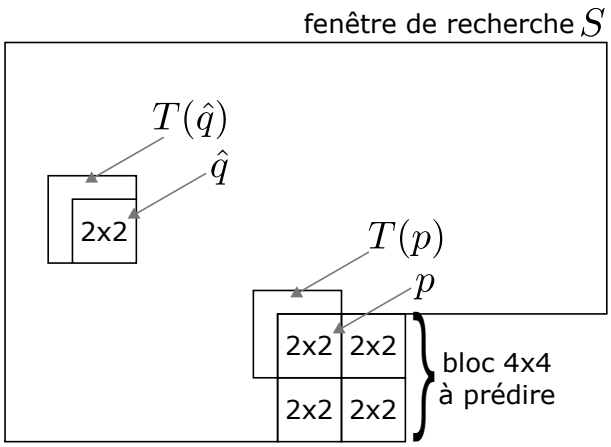


FIGURE 3.12 – Principe de la méthode de prédiction par *template matching* ([137]).

Tan et al. [137] ont élaboré les bases du *template matching* tel qu'il est appliqué dans un contexte de codage vidéo prédictif. Les sections suivantes exposent d'autres travaux qui apportent une amélioration de leur modèle.

3.5.2.2 *Template matching* basé sur la priorité de reconstruction des blocs

Dans l'article [137], l'ordre de parcours des sous-blocs 2×2 au sein d'un bloc à prédire est figé et respecte le mode de parcours classique (*raster scan mode*, de haut en bas et de gauche à droite). Guo et al. [70] proposent une méthode qui adapte le parcours des sous-blocs 2×2 en analysant localement la structure de l'image. Cette approche est directement inspirée de [50] (voir 3.4) qui stipule que pour obtenir une reconstruction visuellement cohérente d'une portion d'image, les contours doivent être restaurés en priorité. La disposition de ces contours est prédite à partir de l'intensité et de l'orientation des contours adjacents à la zone à prédire.

Puisque l'ordre de parcours des sous-blocs est modifié, la taille (en terme de nombre de pixels) et la disposition du *template* varient. Or, le *template matching* utilise le voisinage comme outil de prédiction. La taille et la configuration du *template* influencent donc la recherche du prédicteur $\hat{q}$. Ce constat soulève un inconvénient majeur de cette approche, car en réduisant la taille du *template*, la recherche du meilleur prédicteur est moins pertinente. Les résultats obtenus montrent tout de même un gain moyen de débit de 2% à qualité équivalente par rapport à la méthode de [137]. Les auteurs révèlent également que le pourcentage d'utilisation du nouveau mode de prédiction par *template matching* augmente d'environ 5% par rapport à la méthode précédente.

3.5.2.3 *Template matching* pondéré

Le *template matching*, tel qu'il a été défini à l'origine dans [137], consiste à sélectionner le candidat $T(\hat{q})$ le plus proche du *template* $T(p)$, c'est-à-dire celui qui minimise la SAD entre les deux *templates*. Les travaux de Tan et al. [138] ont apporté diverses améliorations par rapport à la méthode précédente. Dans un premier temps, la largeur des *templates* (la forme en L constituant le voisinage $T(p)$ de p) a été augmentée, et ce en fonction de la taille du bloc à prédire (4×4 ou 8×8). La raison invoquée est que la taille du *template* doit dépendre de la granularité de la texture à prédire : plus

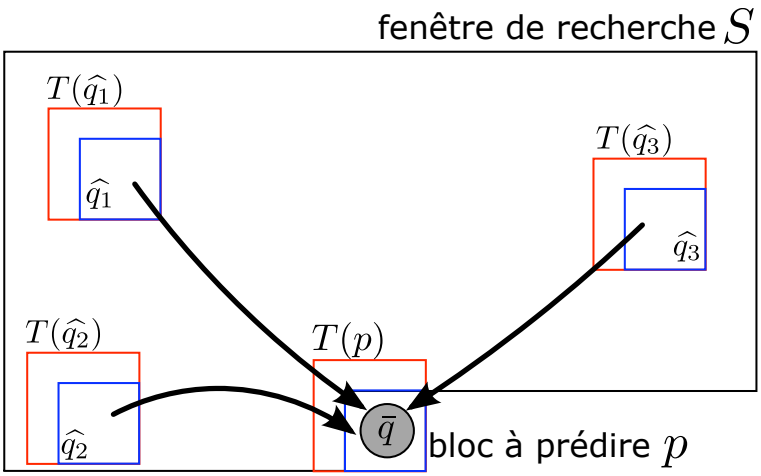


FIGURE 3.13 – Illustration de la méthode de prédiction par *template matching* pondéré, avec $N = 3$.

la taille du *template* est grande et plus la méthode est capable de "capturer" un motif de base de la texture, et donc de prédire correctement le bloc. Cependant, les gains obtenus grâce à cette nouvelle méthode sont faibles.

Le *template* pour un sous-bloc 2×2 est composé des cinq pixels adjacents connus du sous-bloc (Fig.3.12). En s'inspirant des méthodes de prédiction intra-image par extrapolation des pixels adjacents selon une orientation donnée (Fig. 2.6), une nouvelle approche consiste à modifier la configuration spatiale du *template* autour du sous-bloc afin d'exploiter les similitudes selon les directions des structures de l'image. Ainsi, quatre formes de *template* ont été définies, la nouvelle méthode de *template matching* sélectionne la forme qui obtient la SAD minimum (ce qui multiple également par quatre la complexité en terme de calcul de la recherche du meilleur candidat). Avec cet outil, les gains de débit observés sont de l'ordre de moins de 1%.

Enfin, Tan et al. ont constaté que d'autres candidats notés $T(\widehat{q_2}) \dots T(\widehat{q_N})$, triés selon l'ordre croissant de SAD par rapport à $T(p)$, peuvent constituer des prédicteurs $\widehat{q_2} \dots \widehat{q_N}$ aussi bons en terme de compromis débit-distorsion que $\hat{q}$ (renommé $\widehat{q_1}$ pour faciliter la notation). L'amélioration de la méthode proposée consiste à former un prédicteur $\bar{q}$ à partir de la moyenne des N plus proches candidats $\widehat{q_1} \dots \widehat{q_N}$ au sens de leurs *templates* $T(\widehat{q_1}) \dots T(\widehat{q_N})$, comme illustré sur la figure 3.13. Les expérimentations ont démontré qu'avec $N = 4$, le gain de débit par rapport à la méthode de *template matching* de référence augmentait en moyenne de 2% pour un ensemble de huit séquences.

3.5.2.4 Compensation en illumination

Une des limites du *template matching* réside dans sa difficulté à prédire une texture dont l'illumination n'est pas uniforme. En effet, la méthode employée part de l'hypothèse que les régions présentant les même structures ont également la même illumination et le même contraste, ce qui n'est pas toujours le cas pour les séquences naturelles. Pour solutionner ce problème, Zheng et al. [163] propose d'intégrer une compensation en illumination à la méthode de [137] sur la base des travaux de [162]. Le procédé utilise un modèle de compensation linéaire avec des paramètres de *scaling* et d'*offset* pour prendre en compte respectivement les variations de contraste et de luminosité. Ainsi, pour chaque candidat $T(q)$, ces deux paramètres sont calculés par une méthode de résolution des moindres carrés au codeur et au décodeur, et sont ensuite appliqués sur le prédicteur associé q (on part du principe

que l'illumination et le contraste ne varient pas localement entre un template $T(q)$ et le bloc associé q). Une simplification de cette approche consiste à ne pas considérer les variations de contraste, jugées faibles pour des régions adjacentes. Les performances des deux méthodes sont quasi-similaires, et conduisent à une réduction significative de la complexité. Cependant, sous certaines conditions, la méthode proposée réduit les performances de codage (pour les régions très bruitées). La solution consiste donc à signaler par macro-bloc si la compensation en illumination est utilisée ou non. Les gains de débit obtenus par rapport à la méthode de prédiction par *template matching* standard s'élèvent à 2%.

3.5.2.5 *Template matching* appliqué au codage inter-frame

D'autres travaux ont été réalisés dans le cadre de la prédiction inter-frame par *template matching* [132, 135, 136]. Dans ce contexte, l'approche par *template matching* se substitue au codage des vecteurs de mouvement. Dans ce cas, le mouvement est estimé à partir du voisinage causal disponible. L'intérêt du *template matching* est que la méthode opère à la fois au codeur et au décodeur et qu'elle ne nécessite aucun codage d'information supplémentaire. Le *template matching* peut également être utilisé pour affiner le vecteur de mouvement ; la méthode est ainsi complémentaire par rapport à l'approche classique.

3.5.2.6 Conclusion

Nous constatons que les travaux de Tan et al. en 2006 sur le codage vidéo par template matching ont suscité un entrain de la communauté scientifique au vu du grand nombre de contributions qui ont ensuite décliné leur modèle. Cette approche présente plusieurs intérêts. Tout d'abord, les méthodes de *template matching* s'intègrent aisément dans le codeur basé H.264/AVC puisqu'elles conservent un système d'estimation de qualité objective. De ce fait, elles sont compatibles avec les mécanismes d'optimisation débit-distorsion classiques. Cette différence les démarque des approches de codage vidéo par analyse-synthèse, décrites dans la partie 3.5.1, qui prennent en compte des aspects psycho-visuels pour augmenter les performances de codage. Cependant, ce type d'approche ne permet pas de définir un schéma de codage vidéo générique et robuste.

En lien avec la méthode de *template matching*, il existe une autre méthode de prédiction exploitant les propriétés d'autosimilarité d'une image appelée prédiction par déplacement intra-image [162]. L'idée est d'étendre au codage intra le mécanisme de compensation de mouvement. Dans la portion d'image codée-décodée, le bloc qui correspond le plus à la cible est utilisé en tant que prédicteur. Pour reconstruire le bloc au décodeur, un vecteur de déplacement indiquant la position du bloc source dans l'image est transmis. Ce procédé garantit que le prédicteur utilisé par recopie de bloc est le meilleur disponible dans l'image codée. Par rapport à la prédiction par déplacement intra, l'avantage du *template matching* dans un contexte de codage est que la méthode ne nécessite pas d'information transmise supplémentaire puisque le procédé est répliqué au décodeur, ce qui en fait une alternative intéressante par rapport à l'estimation-compensation de mouvement décrit par des vecteurs. Cependant, la méthode de recherche du plus proche *template* dans l'image préalablement codée/décodée alourdit considérablement la complexité de calcul au décodeur. L'article [18] propose d'utiliser le déplacement intra et le *template matching* comme deux nouveaux modes de prédiction intra. Les gains obtenus selon la métrique de Bjontegaard [25] sont de l'ordre de 5% en intra.

3.6 Conclusion

Dans ce chapitre, nous avons exploré plusieurs domaines liés au traitement d'image, en particulier la synthèse de texture, la restauration d'image par synthèse et le codage d'image/vidéo. La synthèse de texture, ainsi que les méthodes de restauration d'image qui en découlent, ont des critères de qualité purement subjectifs. En effet, la capacité d'une méthode à synthétiser/restaurer une image ne peut être mesurée qualitativement. Nous utilisons un *a priori* du rendu psycho-visuel de l'image créée pour estimer l'efficacité de la méthode.

Ces méthodes de synthèse et de restauration, appliquées au contexte de codage vidéo, ont inspiré deux schémas de compression foncièrement différents.

Le premier consiste à séparer les textures qui peuvent être synthétisées sans introduire de gêne visuelle pour les autres éléments de la séquence (codés *via* une technique classique de codage vidéo). L'intérêt est de pouvoir réduire considérablement l'information nécessaire à la reconstruction de ces textures, soit en transmettant un échantillon de texture de taille restreinte, soit en transmettant un ensemble de paramètres statistiques de texture. Cependant, le fait de ne pas pouvoir contrôler avec exactitude la qualité de la synthèse fait que la méthode, très efficace sous certaines conditions, peut produire des résultats désastreux. De plus, les outils de segmentation spatio-temporelle sont très lourds avec en général des résultats sous-optimaux.

La deuxième approche consiste à utiliser un modèle de champ de Markov pour prédire une portion d'image. En caractérisant chaque bloc par son voisinage causal, la méthode cherche à exploiter les similarités au sein même de l'image. Cette approche permet de contrôler la qualité de la prédiction selon des critères objectifs, ce qui est un avantage par rapport au procédé précédent. En effet, ce mécanisme de prédiction peut être directement intégré dans un codeur vidéo de type H.264/AVC, dont les outils d'optimisation débit-distorsion sont reconnus pour leur efficacité. Le chapitre suivant décrit notre contribution sur ce nouveau schéma prédictif de texture.

CHAPITRE 4

Nouvelle méthode de prédiction par *template matching* appliquée au codage vidéo

Sommaire

Ce chapitre est dédié à l'amélioration de la méthode de prédiction par *template matching* dans un contexte de codage vidéo basé bloc type H.264/AVC. Il constitue une première contribution dans le cadre de cette thèse. L'introduction de ce chapitre situe le procédé innovant dans le chaînage des traitements successifs opérés dans un codeur vidéo hybride. Il rappelle ensuite le formalisme de la méthode de *template matching* et décrit les limites des approches précédentes. Puis, nous présentons notre méthode pour répondre aux besoins spécifiques du codage vidéo. Dans la troisième section, avant de conclure, nous reviendrons sur les détails techniques de l'implémentation proposée suivie d'une présentation et d'une discussion des résultats.

4.1 Introduction

4.1.1 *Template matching* et codage vidéo

De façon classique, la méthode de prédiction par *template matching* fournit un outil supplémentaire permettant de prédire dans le domaine spatial un bloc p à partir d'un bloc adéquat $\hat{q}$ appartenant à

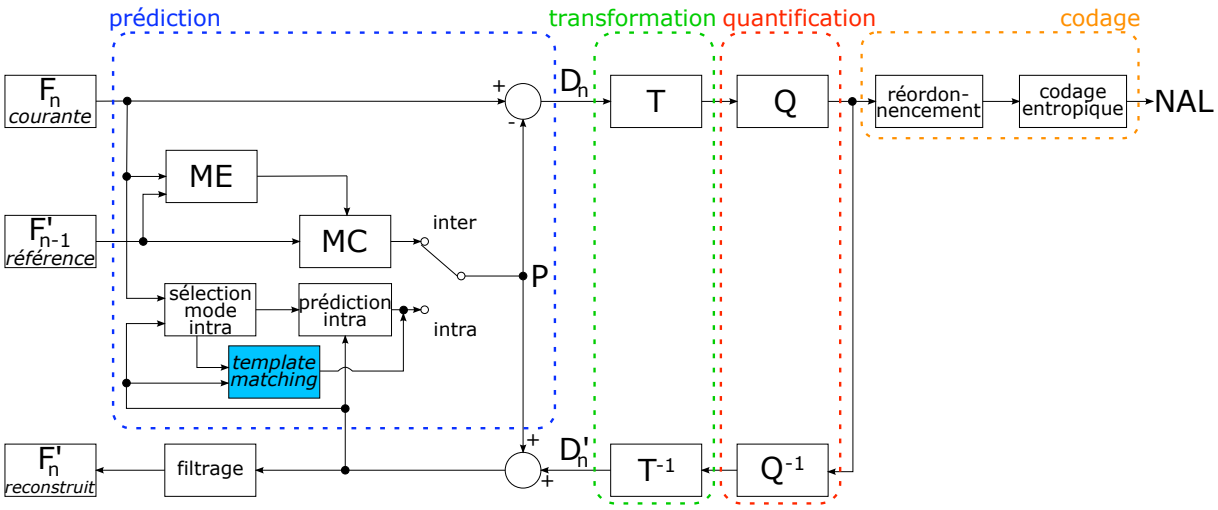


FIGURE 4.1 – Nouvelle méthode de prédiction par *template matching* dans un codeur vidéo type H.264.

une image source S. Dans le cadre du codage vidéo intra-image, l'image source S constitue la portion de l'image déjà codée/décodée située dans l'environnement proche du bloc à prédire. Ainsi, le même procédé de prédiction du bloc peut être effectué de façon identique au codeur et au décodeur. La méthode de prédiction par *template matching* est mise en compétition avec les autres modes de prédiction intra-image proposés dans H.264/AVC. De ce fait, la fonction d'optimisation débit-distorsion *RD-opt*[1] permet de sélectionner la méthode prédictive la plus efficace. L'imbrication de la méthode de prédiction par *template matching* au sein d'un codeur vidéo H.264/AVC est illustrée sur la figure 4.1, où le *template matching* est proposé comme une alternative à la prédiction intra. Là où les autres méthodes de prédiction intra-image de H.264/AVC ne nécessitent que les pixels adjacents du voisinage causal (voir 2.3.2.2) pour calculer le prédicteur par interpolation, la méthode par *template matching* requiert la manipulation d'une image source S de taille non négligeable. Sur la figure 4.1, nous constatons que l'intégration de la nouvelle méthode de prédiction n'interfère pas avec les autres étapes du procédé de codage : transformation, quantification et codage entropique.

Dans la méthode de prédiction par *TM*, à chaque bloc cible p à prédire est associé un *template* noté $T(p)$, composé des pixels adjacents connus (identiques aux pixels utilisés pour les autres modes de prédiction intra-image). Le plus proche template $T(\hat{q})$ de l'image source S fournit le bloc $\hat{q}$ utilisé comme prédicteur de p par recopie des pixels (illustration sur la figure 3.12). En pratique, la notion de distance d entre deux *templates* $T(a)$ et $T(b)$ est exprimée par leurs différences pixel à pixel (ici la *SSD*) :

$$d\left(T(a), T(b)\right) = \sum_{i=1}^{\text{card}(T)} \left(T(a,i) - T(b,i)\right)^2 \tag{4.1}$$

où $T(a,i)$ correspond à la valeur du $i^{ème}$ pixel du *template* $T(a)$, et card(T) au nombre de pixels qui composent un *template*.

1. *Rate-Distortion Optimization*

L'intérêt de la méthode de prédiction par *TM* est qu'elle permet de prédire efficacement des portions d'image qui se répètent. En premier lieu, nous pouvons notamment penser aux textures rigides composées de motifs très similaires présentant peu de variations de géométrie, de couleur et de variation d'illumination. Cependant, nous constatons que cette méthode est aussi couramment employée pour prédire un contour rectiligne possédant une orientation qui ne concorde avec aucune direction d'interpolation des modes de prédiction intra-image de H.264/AVC. Dans ce cas, la répétition d'un échantillon du contour est plus efficace que l'interpolation de ce contour selon une orientation proche de la sienne.

4.1.2 Motivation

La méthode par *template matching* est issue des algorithmes de restauration d'image et de synthèse de texture qui découlent de l'hypothèse qu'un bloc d'image peut être caractérisé par le lien qui l'unit à son voisinage. En recherchant les même redondances dans une donnée source, il est possible de reconstruire une image de façon naturelle en recopiant localement des portions d'images extraites de l'information connue.

Dans le domaine de la synthèse, ce modèle se suffit à lui même, car il consiste à créer une nouvelle image en imitant le caractère aléatoire d'une texture source.

Dans le domaine de la restauration d'image, où l'objectif est de supprimer un objet et donc de propager le fond de l'image sur la région à remplacer, la méthode donne une solution parmi plusieurs plausibles. Cependant, il n'est pas possible avec ce genre de méthode de restaurer un fond composé de plusieurs objets ordonnancés sur des plans différents. Dans ce cas, il est très difficile de savoir, de façon automatique, quel objet en masque partiellement un autre. Ces problèmes complexes font appel à une compréhension globale de la scène, qui n'est pas prise en compte avec ce mécanisme de restauration puisqu'il procède localement. Nous atteignons ici une limite de cette approche et c'est pourquoi la qualité de la reconstruction dépend notamment de la taille des régions et de la simplicité structurelle du fond à restaurer.

Dans le domaine du codage vidéo, le *template matching* est un procédé de prédiction, c'est-à-dire que nous avons une connaissance *a priori* du bloc cible à "restaurer". Dans ce cas, le modèle proposé peut fournir un bloc prédicteur dont le voisinage est le plus proche selon une métrique quelconque, sans toutefois correspondre réellement au bloc cible. Il se peut même que le deuxième, le troisième ou le n-ième prédicteur le plus proche soit meilleur que le premier au sens de la distorsion. Les travaux de Tan et al. [138] sont partis de ce constat. Leur méthode propose de moyenner les n plus proches candidats pour obtenir le prédicteur. Cependant, si les n premiers candidats sont très différents, le fait de les pondérer peut engendrer un prédicteur très éloigné de l'image d'origine. Pour illustrer ce cas, prenons l'exemple d'une image binaire (Fig. 4.2). La moyenne des pixels de deux candidats (en bleu) forment sur la figure un prédicteur (en rouge) composé d'une couleur qui n'apparaît pas dans l'image source, ce qui créer un résultat visuel non naturel. Intégré dans un codeur vidéo, nous nous rendons ainsi compte de l'inefficacité de ce type d'approche dans certains cas.

La méthode que nous proposons part du constat que le n-ième prédicteur par *template matching* peut être meilleur que le premier, de telle sorte qu'il peut être intéressant de transmettre une information supplémentaire pour indiquer au décodeur quel candidat de la liste des n meilleurs est utilisé pour prédire le bloc cible.

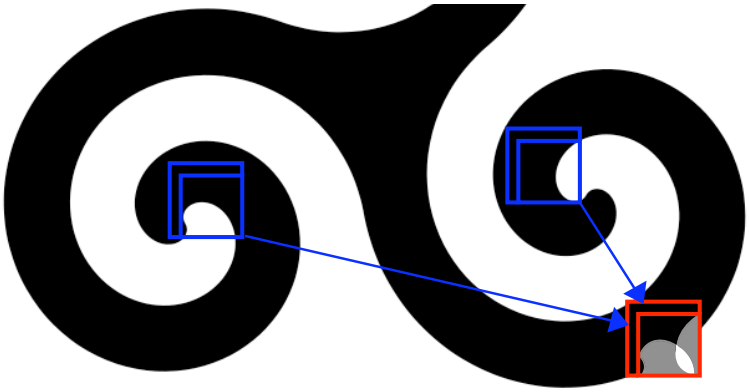


FIGURE 4.2 – Illustration d'un échec de la méthode de prédiction par *template matching* pondéré (ATM). La cible (en rouge) est calculée en moyennant les deux meilleurs candidats (en bleu).

4.2 Schéma prédictif adapté par *template matching*

La méthode de *template matching* à candidats multiples (STMP pour "Set of Template Matching Predictors") que nous proposons se décompose en deux étapes :

Etape 1 : construction de la liste ordonnée des N meilleurs candidats (au sens du *template*) ;

Etape 2 : codage/décodage de l'indice du meilleur prédicteur (au sens du bloc cible).

La première étape est similaire à l'encodeur et au décodeur pour obtenir une liste $\mathcal{L}_N$ rigoureusement identique. Au contraire, la différenciation au niveau de la deuxième étape concerne la sélection côté encodeur entre tous les candidats disponibles. Dans ce cas, il est nécessaire d'intégrer un traitement supplémentaire qui permet de sélectionner le prédicteur optimal. Au décodeur, il suffit d'utiliser l'indice transmis et la liste $\mathcal{L}_N$ pour obtenir le prédicteur.

4.2.1 Etape 1 - construction de la liste des N meilleurs candidats

La source S est constituée de la portion de l'image précédemment codée/décodée. Dans cette source, nous recherchons les N meilleurs candidats $\widehat{q_1}...\widehat{q_N}$ dont les *templates* $T(\widehat{q_1})...T(\widehat{q_N})$ minimisent la distance par rapport au template $T(p)$ du bloc cible p (Eq. 3.8 et 4.1). Ces candidats sont stockés dans une liste $\mathcal{L}$ triée par ordre croissant de la distance d du template candidat au template cible, et telle que :

$$\mathcal{L}_i = \underset{T(q)\in S/\mathcal{L}_h}{\arg\min}\, d(T(p),T(q)),\ i,h \in [1,i[\tag{4.2}$$

où $\mathcal{L}_i$ représente le $i^{\text{ème}}$ candidat le plus proche de $T(p)$ au sens de la fonction de distance $d(\cdot)$.

La liste $\mathcal{L}$ constitue l'ensemble de nos prédicteurs disponibles (illustration Fig. 4.3).

4.2.2 Etape 2 - codage/décodage de l'indice du meilleur prédicteur

Pour cette étape, nous reprenons les mécanismes utilisés dans H.264/AVC. Au cours de l'encodage, la fonction débit-distorsion RD-opt permet d'extraire le prédicteur $\widehat{q_i}\ \forall i \in [1,N]$ de la liste $\mathcal{L}_N$ qui

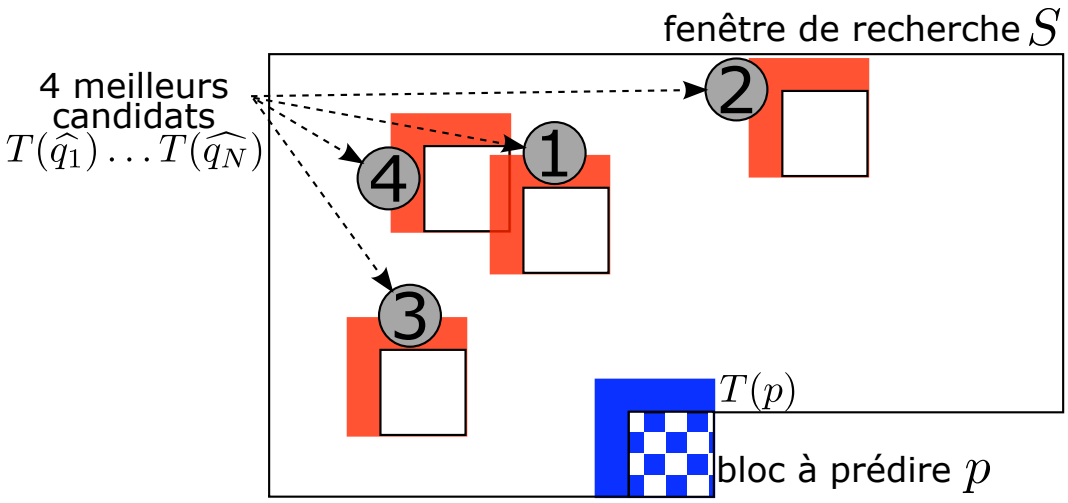


FIGURE 4.3 – Construction de la liste $\mathcal{L}_N$ des N candidats les plus proches du template cible $T(p)$ ($N = 4$).

offre le meilleur compromis. Le coût de codage est mesuré en prenant en compte la signalisation de l'indice i indiquant la position de prédicteur $\widehat{q_i}$ dans la liste $\mathcal{L}_N$ et le coût de codage du résidu (l'erreur de prédiction). L'indice i est signalé dans le flux pour que le décodeur, disposant de la même liste $\mathcal{L}_N$, puisse effectuer l'opération de prédiction (Fig. 4.4).

4.2.3 Discussion

1. Le fait de trier les éléments de $\mathcal{L}_N$ se justifie si l'on considère que l'indice du meilleur candidat i est codé par un codeur entropique. En effet, nous supposons que la distribution de i n'est pas équiprobable, mais qu'il dépend de la distance $d(\cdot, \cdot)$ par rapport au *template*. En d'autres termes, plus un candidat est proche du *template*, plus il a de chance d'être le meilleur prédicteur.
2. Le paramètre N, la taille de la liste, a un rôle important ; il définit le nombre de prédicteurs et le coût de codage de l'indice i. Plus N est grand et plus le coût de codage de i est élevé (coût de codage d'un symbole parmi N), mais plus il y a de probabilité d'obtenir le meilleur prédicteur $\widehat{q_i}$ possible de l'image source S. Il peut être statique sur l'ensemble du processus de codage, ou dynamique pour s'adapter au contenu de l'image. Nous avons mesuré expérimentalement la taille optimale N pour coder des séquences naturelles.
3. Le *template matching*, sous sa forme basique, procède par recopie de blocs. Nous pouvons intégrer d'autres outils permettant de générer des prédicteurs à partir de l'image source, telles que la recherche de candidats au demi et au quart de pixel, la compensation en illumination, les opérations géométriques (rotation, homothétie, . . .), la construction d'un prédicteur à partir de plusieurs candidats, etc.

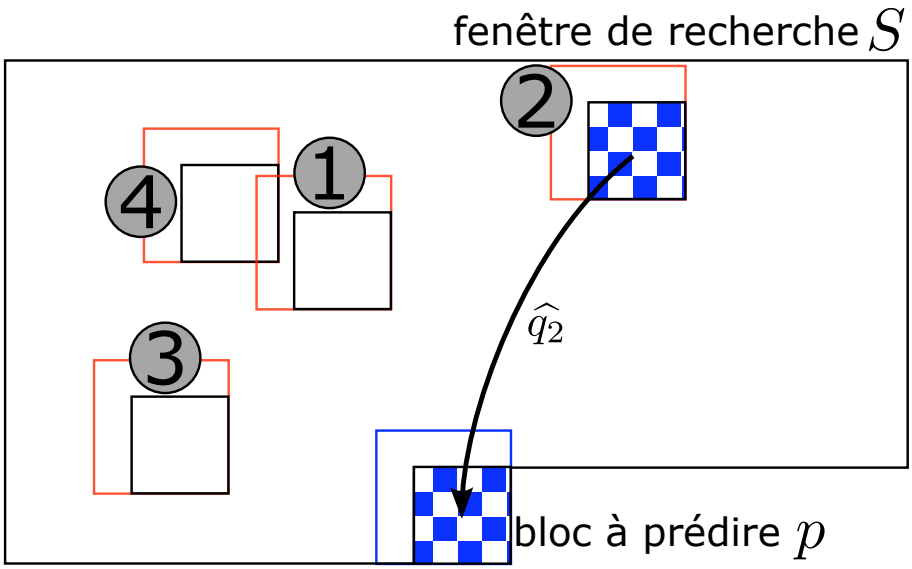


FIGURE 4.4 – Choix du meilleur prédicteur $\widehat{q_i}$ de la liste des N meilleurs candidats (avec $i = 2$).

4.3 Expérimentations et résultats

Au cours du procédé de prédiction intra-image du standard H.264/AVC, l'image est décomposée en blocs 4×4, 8×8 ou 16×16 pixels (voir 2.3.2). Les techniques de codage étant légèrement différentes, il nous a semblé judicieux d'adapter le *template matching* en fonction de la taille des blocs traités. Nous nous focaliserons sur les blocs de taille 4×4 et 8×8 pixels.

4.3.1 Détails techniques de l'implémentation

Il existe neuf modes de prédiction intra-image dans H.264/AVC (figure 2.6) ; le mode *template matching* y est ajouté. Pour que le coût de codage du nouveau mode $STMP$ n'influe pas sur les performances du codeur, il convient de modifier la manière dont sont codés les modes de prédiction intra-image. Ensuite, nous exposerons les détails techniques de l'algorithme du *template matching* que nous avons expérimenté.

4.3.1.1 Signalisation d'un nouveau mode de prédiction intra-image

Dans H.264/AVC, une fonction nommée MPM[2] est utilisée pour estimer le mode de prédiction d'un bloc à partir de ses blocs voisins. En effet, il est communément admis qu'il y a une corrélation des modes de prédiction utilisés entre blocs voisins, basé sur l'hypothèse que les structures d'image se répètent d'un bloc à l'autre. Soit un bloc C entouré d'un bloc A et d'un bloc B préalablement codés, comme illustré sur la figure 4.5. Aux blocs A et B correspondent un mode de prédiction intra-image noté respectivement a et b ($a < 0$ et $b < 0$ si A et B ne sont pas disponibles respectivement). La fonction MPM est définie comme suit :

2. *Most Probable Mode*

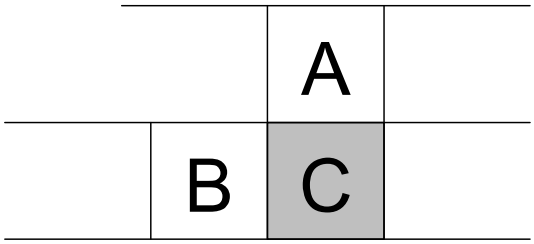


FIGURE 4.5 – Estimation de mode de prédiction intra-image du bloc C à partir de ses blocs voisins A et B.

$$MPM(a,b) = \begin{cases} mode\ DC & si \quad a < 0\ ou\ b < 0 \\ \min(a,b) & sinon \end{cases} \tag{4.3}$$

Un symbole binaire S_{MPM} indique le résultat de l'équation :

$$S_{MPM} = \begin{cases} vrai & si \quad c = MPM(a,b) \\ faux & sinon \end{cases}, \tag{4.4}$$

avec c correspondant au mode de prédiction du bloc C. Le symbole binaire S_{MPM} est transmis. Si S_{MPM} est vrai, alors le mode de prédiction c est obtenu directement par la fonction MPM. Dans le cas contraire, il est nécessaire de coder le mode de prédiction c, correspondant à un symbole parmi huit (ensemble constitué des neuf modes de prédiction disponibles à l'exception du mode obtenu par $MPM(a,b)$), ce qui revient à coder trois symboles binaires.

Si nous rajoutons un nouveau mode de prédiction intra-image et que nous conservons le même mécanisme, il restera un symbole parmi neuf à coder dans le cas où $c \neq MPM(a,b)$, ce qui nécessite quatre bits à transmettre et rend donc la méthode de codage sous optimale. Pour éviter ce cas de figure, Guo et al. [70] proposent une solution qui consiste à coder à la suite de S_{MPM} un symbole binaire supplémentaire pour indiquer si le bloc C est prédit par *template matching*. S'il ne l'est pas, il ne reste alors que huit possibilités ; nous revenons donc au cas de figure classique. Cette méthode n'est pas optimale. En effet, avec ce mécanisme, la signalisation du mode TM est en moyenne moins coûteuse à coder ; il est donc favorisé vis-à-vis des autres modes.

La solution que nous proposons consiste à utiliser une seconde fonction de type MPM appelée $SMPM$ [3]. Le mode de prédiction par *template matching* est associé au numéro neuf. Les fonctions MPM et $SMPM$ sont donc déclinées ainsi :

$$MPM(a,b) = \begin{cases} mode\ DC & a < 0\ ou\ b < 0 \\ \min(a,b) & sinon \end{cases} \tag{4.5}$$

$$SMPM(a,b) = \begin{cases} mode\ TM & a < 0\ et\ b < 0 \\ \max(a,b) & a \neq b \\ mode\ DC & else \end{cases} \tag{4.6}$$

Comme le symbole S_{MPM}, le symbole binaire S_{SPMP} est vrai si $c = SPMP(a,b)$. Si S_{SPMP} est faux, alors un symbole parmi les huit restant est codé. L'imbrication des méthodes MPM et $SPMP$ est

3. *Second Most Probable Mode*

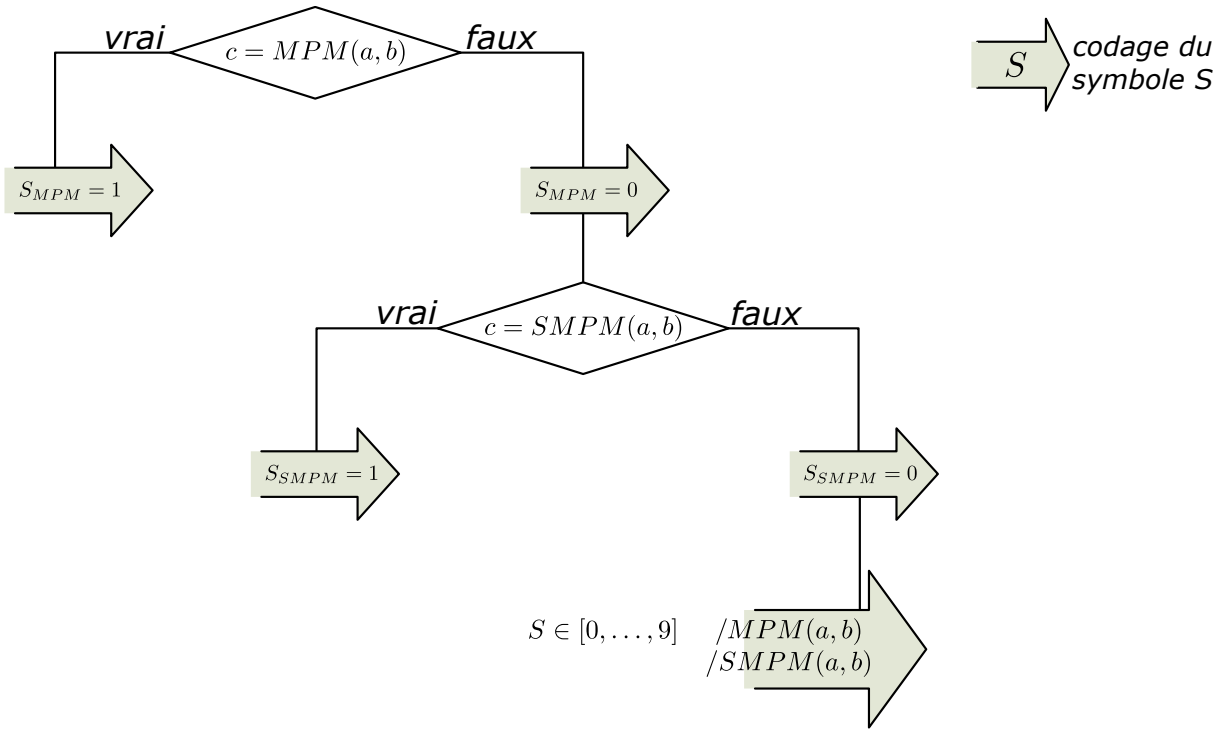


FIGURE 4.6 – Schéma de codage proposé du mode de prédiction c parmi un ensemble de dix modes.

illustrée sur la figure 4.6. Ainsi, il y a une équité entre les modes de prédictions intra-image en terme de coût de signalisation. La fonction $SPMP$ est définie de telle sorte que $MPM(a,b) \neq SPMP(a,b)$ afin de pouvoir discriminer deux modes de prédiction intra-image en appelant successivement MPM et $SPMP$. L'utilisation de $\max(a,b)$ dans la fonction $SPMP$ garantit toutefois une concordance avec les blocs voisins de C.

4.3.1.2 Implémentation du template matching à candidats multiples

Le bloc cible étant de taille 4×4 ou 8×8 pixels, nous avons défini les *templates* comme illustré sur la figure 4.7. Pour les blocs 4×4, le template est constitué de dix pixels appartenant au voisinage causal (s'ils sont disponibles), c'est-à-dire des pixels adjacents à une distance d'un pixel du bloc. Le *template* pour les blocs 8×8 est construit à partir des dix-huit pixels voisins. La littérature évoque d'autres types de *templates*. Cependant, les meilleurs résultats ont été obtenus avec ceux illustrés sur la Fig. 4.7.

La fenêtre de recherche dans l'image source S est de taille 32×32 et 64×64 pixels pour les blocs respectivement de taille 4×4 et 8×8. La fenêtre de recherche est centrée sur le pixel en haut à gauche du bloc cible.

Lorsque le mode STMP est sélectionné, l'indice i du meilleur prédicteur $\widehat{q}_i$ de la liste $\mathcal{L}_N$ est codé/décodé. Pour faciliter l'implémentation, la longueur N de la liste est prédéfinie pour toute la séquence, N étant de préférence une puissance de deux. Ainsi, la binarisation de i donne un code binaire de longueur fixe optimal. Nous allons voir dans la section suivante l'impact de la valeur de N sur les résultats.

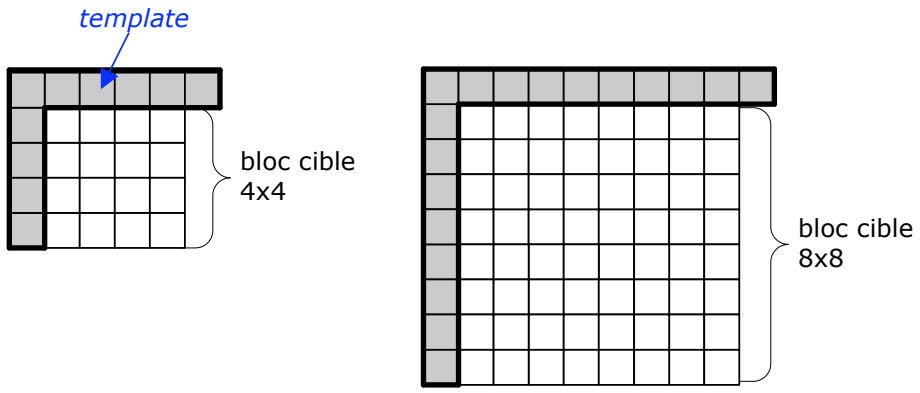


FIGURE 4.7 – Configuration des templates pour des blocs 4×4 et 8×8 pixels.

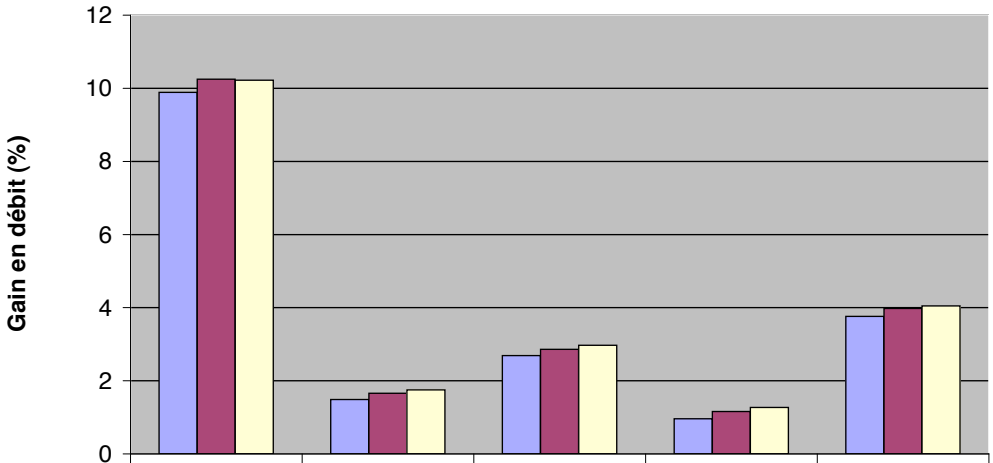
4.3.2 Résultats

Les résultats sont présentés en prenant en compte la métrique proposée par Bjontegaard [25], qui exprime le gain moyen pour différents débits, soit sous la forme d'une réduction de débit, soit d'une augmentation de PSNR (ici, les résultats sont exprimés en terme de gain de débit). La métrique de Bjontegaard nécessite plusieurs débits différents par séquence. Nous utilisons les QP 22, 27, 32 et 37 correspondant à des contextes de codage réalistes (le QP définit le pas de quantification utilisé). La référence est le codeur JSVM H.264/AVC FRext [125]. La DCT par bloc 8×8 est activée. Le codeur entropique utilisé est CABAC. Toutes les images de la séquence sont codées en intra. Nous avons utilisé des séquences de référence couramment employées dans la littérature pour réaliser ces tests, et nous encodons les cinquante premières images de chaque séquence.

4.3.2.1 Incidence de la taille de la liste $\mathcal{L}_N$

Dans un premier temps, nous cherchons à estimer expérimentalement la taille N optimale de la liste pour coder des séquences naturelles. L'histogramme 4.8 montre les gains de débit pour trois tailles de liste 8, 16 et 32 qui offrent les meilleurs résultats dont nous disposons. Tout d'abord, nous constatons que les résultats sont très différents d'une séquence à l'autre. Par exemple pour $N = 32$, la méthode proposée engendre un gain de plus de 10% pour la séquence *foreman*, contrairement à la séquence *tempête* qui plafonne à 1.27% de gain en débit à qualité équivalente. Cette observation rejoint les remarques faites pour les autres méthodes de *template matching*, à savoir que la méthode de prédiction proposée s'adapte mieux à certains types de contenu, en fonction des structures/textures qui composent l'image.

Dans un deuxième temps, bien que les résultats soient très similaires en fonction de N, nous obtenons pourtant une différence notable de coût de signalisation. En effet, nous passons de trois bits par bloc pour $N = 8$ à cinq bits pour $N = 32$ (ce qui peut représenter plus de 12 kbits de différence pour une image CIF). En moyenne, les résultats sont meilleurs pour $N = 32$. Nous allons utiliser cette configuration pour les tests suivants.



	foreman	mobile	paris	tempete	MOYENNE
N=8	9,89	1,49	2,69	0,96	3,76
N=16	10,25	1,66	2,86	1,16	3,98
N=32	10,22	1,75	2,97	1,27	4,05

FIGURE 4.8 – Influence de la taille N de la liste pour la méthode STMP sur des séquences CIF (réf : H.264/AVC FRext)

4.3.2.2 Comparaison avec les méthodes antérieures de la littérature

Pour évaluer notre méthode, nous comparons nos résultats avec ceux des méthodes de l'état de l'art. Pour cela, nous avons sélectionné la méthode de prédiction par *template matching* (TM) de Tan et al. et présentée dans 3.5.2.1. Nous prenons également comme référence la méthode de template matching pondérée (ATM^4) avec $N = 4$, introduite dans 3.5.2.3, et qui donne à ce jour les meilleurs résultats pour ce type d'approche. Les résultats sont illustrés Fig. 4.9. Nous constatons que notre méthode est en moyenne meilleure pour les séquences CIF, alors qu'à l'inverse, la méthode ATM surpasse de 0.3% la méthode STMP pour les séquences 720p. Quoiqu'il en soit, les résultats entre les deux approches ATM et STMP sont très proches dans ce cas de figure, alors que la méthode classique TM est moins performante (-1.45% en CIF et -1.19% en 720p en moyenne par rapport à notre méthode). Au final, **la méthode STMP** présente une réelle amélioration par rapport à H.264/AVC car elle réduit le débit de façon significative (de **3.19**% globalement).

4.3.2.3 Introduction de la recherche au demi-pixel

Afin d'augmenter le nombre et la précision de candidats potentiels $\widehat{q}_i$, nous introduisons la recherche de *template* dans l'image source S interpolée au demi-pixel, à l'image de ce qui existe déjà pour le procédé d'estimation de mouvement dans le contexte de prédiction inter-images. Nous procédons à une interpolation deux à deux ligne/colonne des pixels de l'image source S, pour générer trois images supplémentaires (issues d'une interpolation horizontale, verticale et des deux cumulées).

Dans un premier temps, nous souhaitons vérifier si la taille N est toujours optimale pour $N = 32$ lorsque nous introduisons la recherche au demi-pixel. La figure 4.10 illustre les résultats obtenus en terme de gain de débit par rapport à H.264/AVC FRext. Nous constatons que non seulement

4. *Averaged Template Matching*

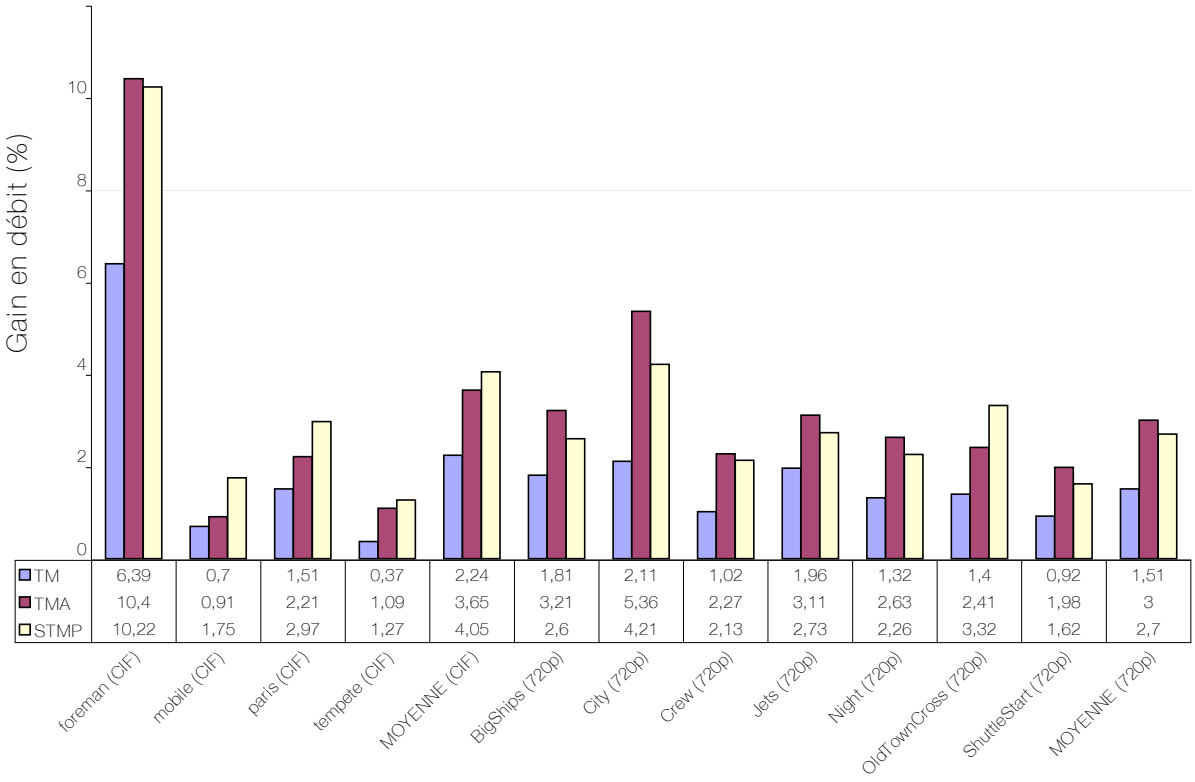
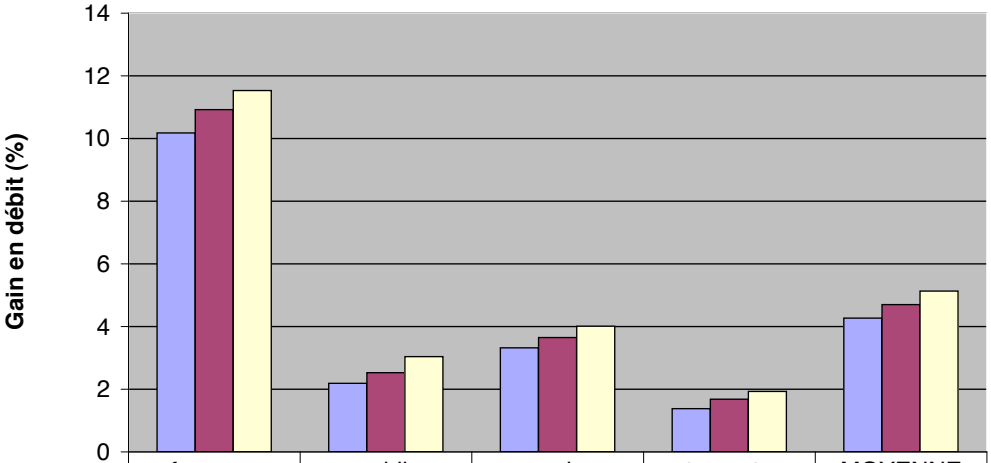


FIGURE 4.9 – Comparatif de la méthode de prédiction TM[137], ATM [138] ($N = 4$) et $STMP$ (pour $N = 32$). Réf : H.264/AVC FRext.



	foreman	mobile	paris	tempete	MOYENNE
N=8	10,18	2,19	3,32	1,38	4,27
N=16	10,92	2,53	3,65	1,68	4,7
N=32	11,53	3,04	4,01	1,93	5,13

FIGURE 4.10 – Influence de la taille N de la liste pour la méthode STMP sur des séquences CIF (réf : H.264/AVC FRext), avec recherche des meilleurs candidats au demi-pixel.

les meilleurs résultats sont obtenus avec $N = 32$, mais également que l'écart se creuse entre les différentes tailles N par rapport à la Fig. 4.8. Au final, avec $N = 32$, la recherche au demi-pixel réduit le débit de 1.08% par rapport à la méthode STMP sans le demi-pixel, et respectivement de 0.94% et 0.51% pour $N = 16$ et $N = 8$.

Dans la deuxième série de tests, nous comparons à nouveau la méthode STMP avec les autres méthodes de la littérature. La recherche au demi-pixel est activée pour toutes les méthodes (TM, ATM et STMP). Il est intéressant de noter plusieurs points :

- L'introduction de la recherche au demi-pixel de la méthode STMP est profitable puisqu'elle réduit globalement le débit de 1.33% par rapport à la méthode sans recherche au demi-pixel.
- Le constat est identique pour la méthode TM classique ; la réduction de gain est de l'ordre 0.77%.
- Au contraire, la recherche au demi-pixel n'a en moyenne pas d'impact sur les résultats de la méthode ATM. Nous attribuons cela au fait que la méthode ATM, qui procède à un moyennage des prédicteurs, introduit un effet de "lissage" du bloc identique à la méthode d'interpolation que nous utilisons. Le fait d'employer conjointement ces deux méthodes n'améliore pas la qualité de prédiction.

Au final, ces derniers tests ont mis en lumière l'intérêt de la méthode STMP par rapport à l'état de l'art. D'après nos tests, le gain en débit par rapport à H.264/AVC FRext pour l'ensemble des séquences est évalué à 4.52% en moyenne, avec un maximum de 11.53% atteint pour la séquence *foreman*.

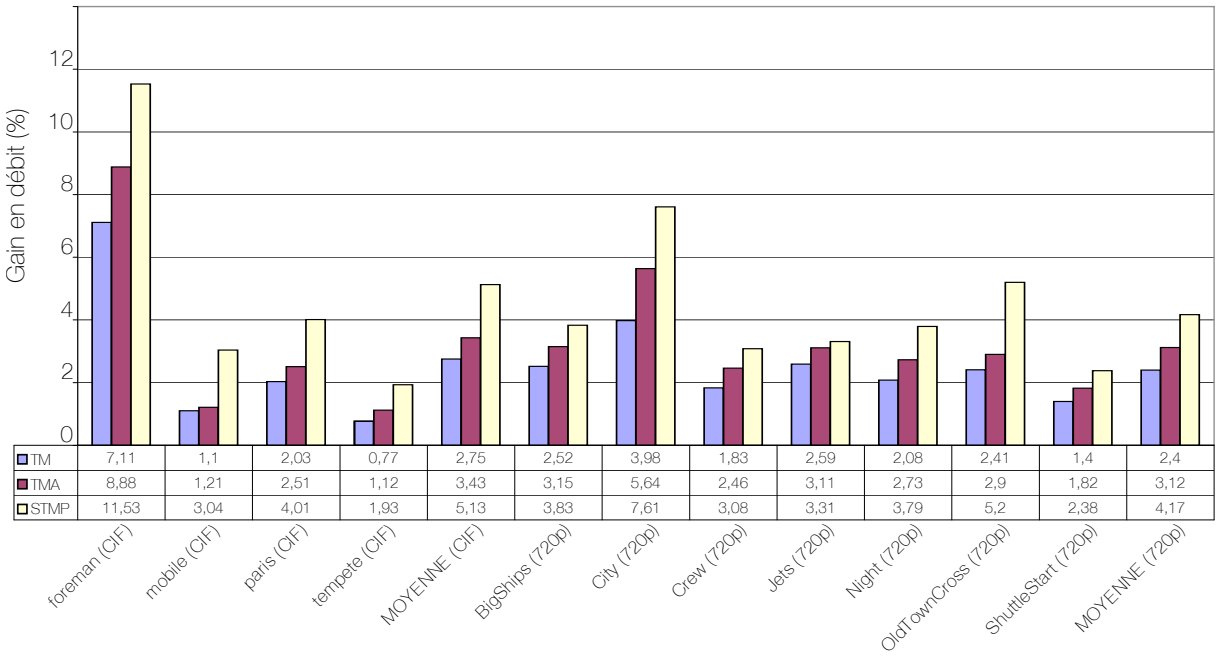


FIGURE 4.11 – Comparatif de la méthode de prédiction *TM*, *ATM* ($N = 4$) et *STMP* (pour $N = 32$) avec recherche au demi-pixel. Réf : H.264/AVC FRext.

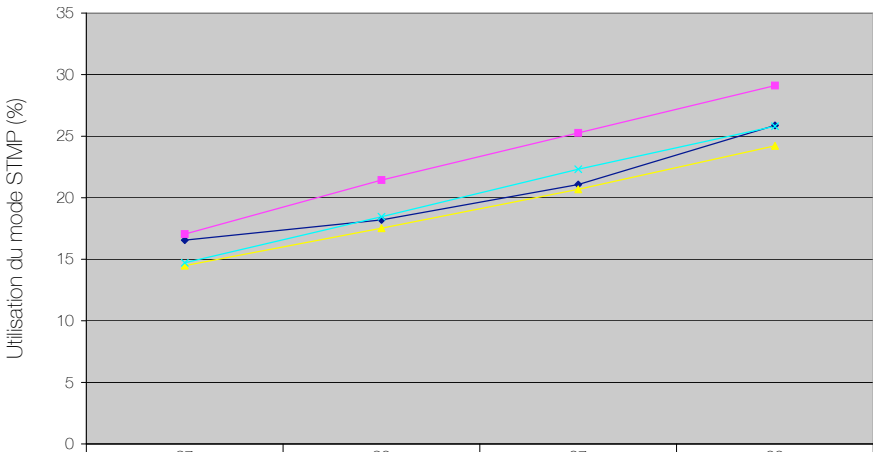
4.3.3 Discussion

4.3.3.1 Utilisation du mode STMP

Nous allons maintenant analyser plus en détails les résultats obtenus grâce à la méthode STMP pour $N = 32$. Le graphique 4.12 montre le taux d'utilisation du mode STMP par rapport aux autres modes de prédiction intra-image pour les blocs 4×4 et 8×8 confondus. Tout d'abord, nous constatons que le mode STMP est employé aussi fréquemment pour les quatre séquences testées, et ce malgré des résultats en termes de performance très différents. Ensuite, l'utilisation du mode STMP varie de façon similaire pour toutes les séquences en fonction du débit. En l'occurrence, le mode STMP est moins utilisé dans des configurations de codage à bas-débit. Cela s'explique par le fait qu'à bas-débit, l'image source S utilisée par la méthode STMP est fortement dégradée, la qualité de la prédiction par *template matching* est donc moins bonne, ce qui favorise au final les autres modes de prédiction intra-image. Au final, avec un taux d'utilisation d'environ 20% en moyenne toutes séquences et débits confondus, le mode STMP est utilisé environ deux fois plus que n'importe quel autre des neuf modes de prédiction intra-image de H.264/AVC, ce qui montre l'efficacité de la méthode.

4.3.3.2 Coût de codage de la signalisation et des résidus

Avec $N = 32$, il faut coder cinq symboles binaires par bloc 4×4 ou 8×8 lorsque le mode STMP est choisi. Sur l'histogramme de la Fig. 4.13, nous différencions le coût de codage de la signalisation, c'est-à-dire l'information nécessaire pour signaler le mode de prédiction et l'indice utilisé pour la méthode STMP, avec le coût de codage du résidu du bloc (en moyenne pour quatre séquences CIF), en fonction de la taille du bloc et du mode de prédiction choisi. Lorsque le mode STMP est utilisé, le



	37	32	27	22
FOREMAN	16,56	18,2	21,07	25,87
MOBILE	17,05	21,43	25,27	29,11
PARIS	14,49	17,52	20,69	24,22
TEMPETE	14,72	18,46	22,31	25,81

FIGURE 4.12 – Taux d'utilisation du mode STMP ($N = 32$) en fonction du QP par rapport aux autres modes de prédiction intra-image sur des séquences CIF.

coût de la signalisation est plus élevé, à cause de la nécessité de coder l'indice i du meilleur candidat. En revanche, nous constatons que le coût de codage du résidu est beaucoup plus faible que pour les modes de prédiction de H.264/AVC. Lorsque les deux sont cumulés, la méthode STMP s'avère en moyenne plus rentable que les autres modes, aussi bien pour les blocs 4×4 que 8×8.

4.3.3.3 Probabilité d'apparition des indices i

La figure 4.14 montre la probabilité d'apparition des indices des meilleurs candidats. Les tests ont été réalisés sur quatre séquences CIF. Cela permet de mettre en avant deux conclusions. Tout d'abord, la distribution n'est pas équiprobable : les indices les plus faibles de la liste, qui correspondent donc aux *templates* les plus proches (au sens de la SSD), sont en moyenne plus souvent sélectionnés (plus de 10%). Cela confirme en partie l'hypothèse de départ qui a donné lieu à la prédiction par *template matching*, à savoir qu'il y a bien une corrélation entre un *template* et le bloc associé qui peut être utilisée pour améliorer le codage des images. Cependant, tous les indices sont utilisés par la méthode STMP, même si les indices les plus élevés apparaissent dans une moindre mesure. Cette constatation contrebalance la première conclusion : dans un codeur d'image basé sur des métriques de distorsion objectives, nous ne pouvons affirmer avec certitude que le meilleur *template* donnera le meilleur prédicteur. La taille $N = 32$ semble donc être un bon compromis entre coût de codage de l'indice et qualité de la prédiction.

4.3.3.4 Evolution en fonction du débit

Enfin, la figure 4.15 illustre l'utilisation de la méthode de prédiction STMP pour une même image à différents QP. A bas-débit (QP 37), il y a beaucoup plus de blocs 8×8 prédits par STMP, ceci afin de mutualiser le coût de signalisation par macro-bloc, alors qu'à haut-débit (QP 22) le partitionnement en blocs 4×4 prédits par la méthode STMP est beaucoup plus utilisé. La figure montre que la méthode STMP est employée surtout pour les zones de contours de l'image.

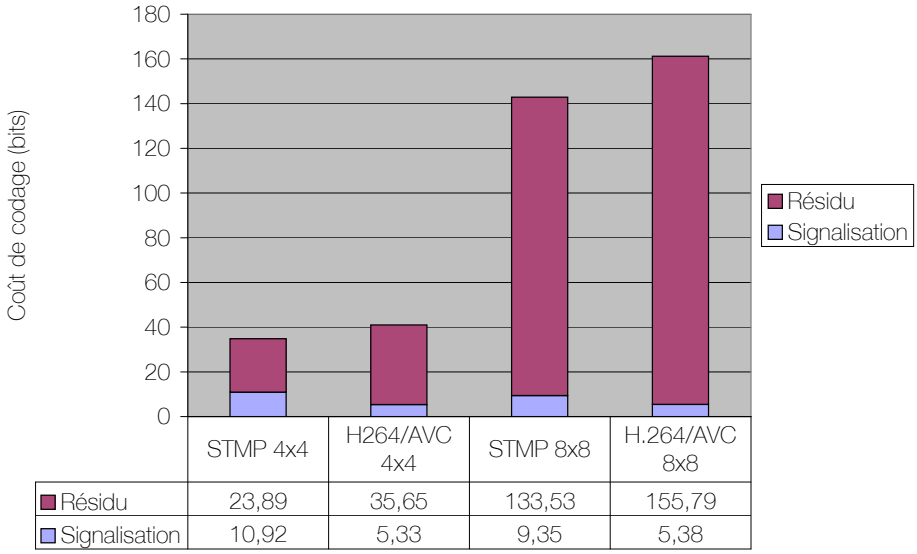


FIGURE 4.13 – Coût de codage moyen (en bits) de la signalisation et du résidu en fonction du mode de prédiction et de la taille du bloc.

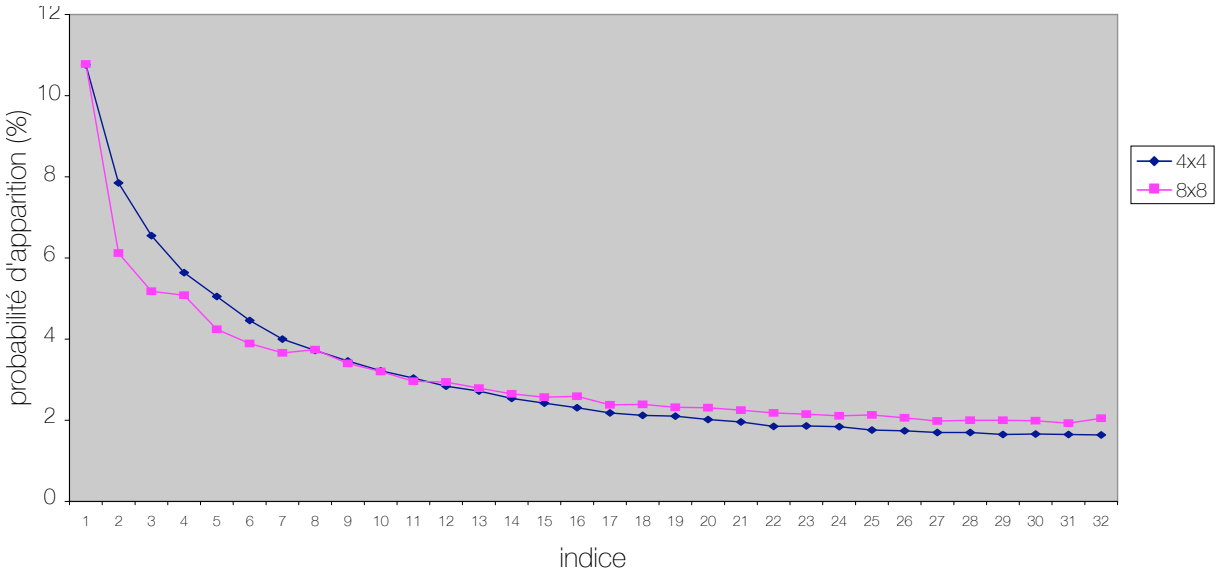
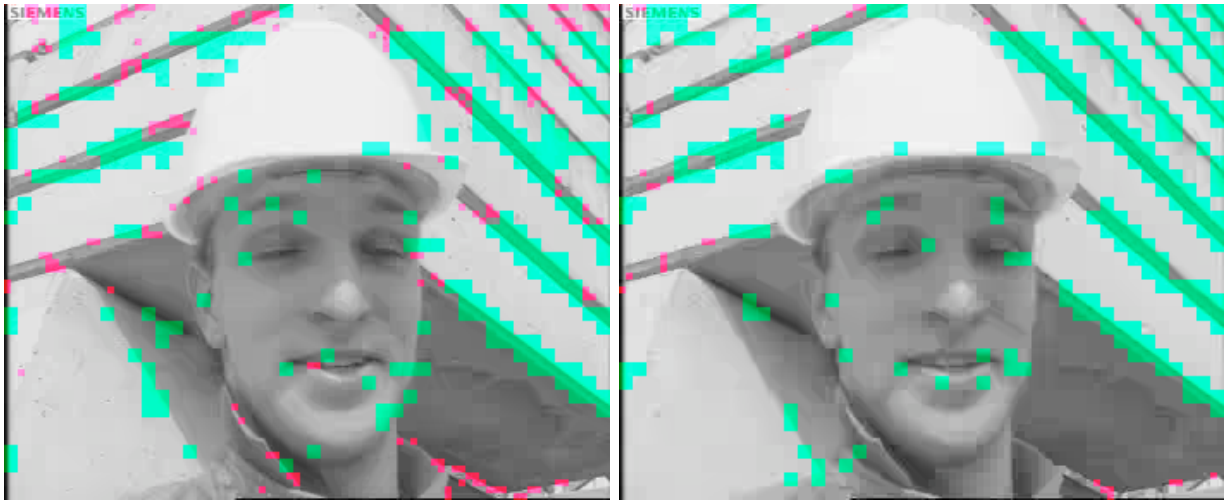


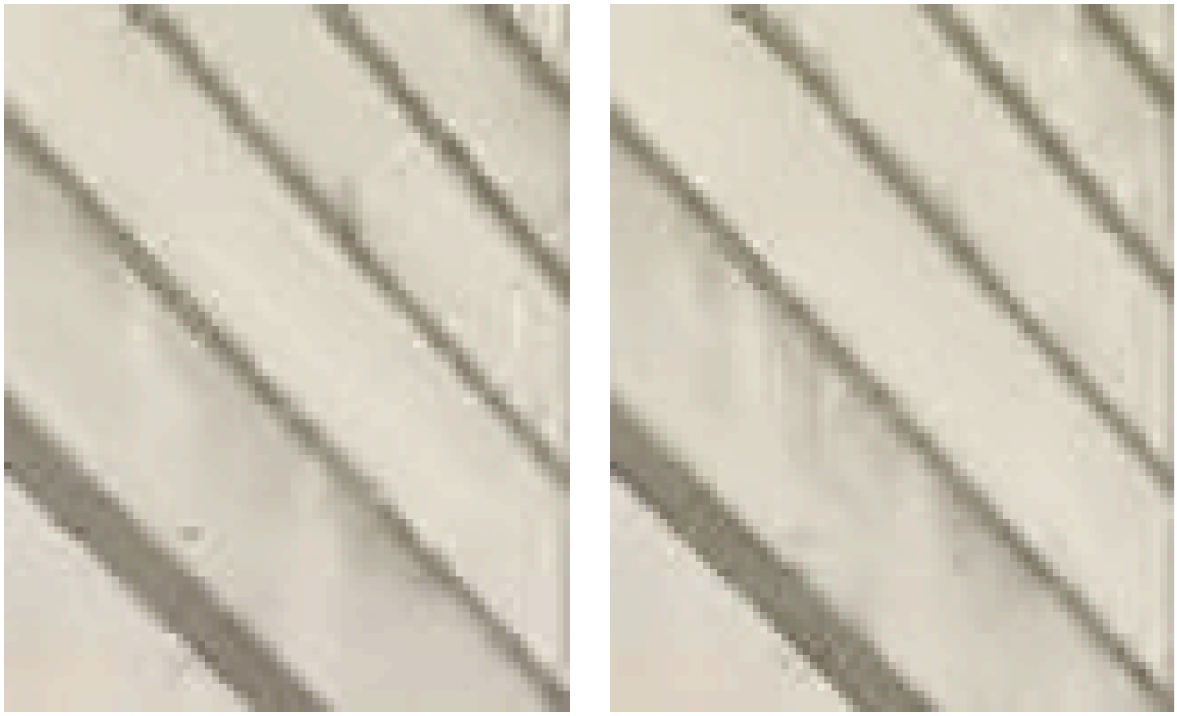
FIGURE 4.14 – Probabilité d'apparition (%) en fonction de l'indice i de la liste $\mathcal{L}_N$ ($N = 32$) et de la taille des blocs.

(a) QP22 (b) QP27



(c) QP32 (d) QP37

FIGURE 4.15 – Illustration des blocs 4×4 (rouge) et 8×8 (vert) prédits par la méthode STMP sur la première image de la séquence *foreman* pour différents débits.



(a) H.264/AVC FRExt. (b) H.264/AVC FRExt + STMP.

FIGURE 4.16 – Différence perceptuelle d'un extrait de la séquence *foreman* avec et sans la méthode STMP.

Pour illustrer l'amélioration perceptuelle apportée, la figure 4.16 compare deux images encodées à débit équivalent et extraites de la séquence *foreman*, avec et sans la méthode STMP activée. Nous nous plaçons dans un contexte de codage à très bas-débit, de l'ordre de 450 kbits/s (en codage intra-image exclusivement), pour volontairement accentuer les artefacts dûs à l'algorithme de compression. Nous constatons au final que la méthode de prédiction STMP permet d'obtenir une image dont les contours sont plus lisses et dont la qualité visuelle est améliorée.

4.4 Conclusion

L'approche que nous avons présentée consiste à proposer plusieurs prédicteurs par bloc 4×4 ou 8×8, tous issus de la méthode de *template matching*. Pour rappel, cette méthode consiste à prédire une région par analyse de son voisinage. En partant de l'hypothèse qu'une image est constituée de motifs quasi-identiques, la similarité entre *templates* souligne une ressemblance entre leurs blocs associés. Cependant, ce principe n'est pas toujours vérifié et il est parfois plus intéressant de prendre en considération les N plus proches candidats plutôt qu'uniquement le premier. Notre méthode transmet l'indice du meilleur prédicteur parmi ces N candidats. Même en prenant en compte ce coût de codage supplémentaire, notre méthode est plus performante que l'état de l'art, avec une réduction de débit de **4.52% en moyenne par rapport à H.264/AVC FRext pour un codage intra-image**.

Sur le principe, nous situons la méthode STMP entre le *template matching* et la méthode de déplace-

ment intra-image ([162, 80] et [18] pour un comparatif des deux méthodes). Cette dernière consiste à transmettre un vecteur de déplacement pour indiquer quel bloc de l'image source est utilisé en tant que prédicteur. Bien que toutes ces méthodes prédisent un bloc à partir d'un autre bloc de l'image, la méthode par *template matching* a l'avantage de ne nécessiter aucune information supplémentaire. Ainsi, le même traitement de recherche du meilleur prédicteur est réalisé au codeur comme au décodeur. Cependant, il n'est pas garanti que le prédicteur choisi par la méthode de *template matching* soit le meilleur possible de l'image source. A l'inverse, la méthode de déplacement intra-image va permettre d'obtenir le meilleur prédicteur possible de l'image, mais au prix d'un coût de codage du vecteur prohibitif. La méthode STMP est entre les deux : le fait de choisir judicieusement N candidats augmente considérablement les probabilités d'avoir le meilleur prédicteur possible, tout en ayant un coût de codage limité. Pour cela, la taille N de la liste est un paramètre primordial que nous avons défini expérimentalement à 32. En réduisant N, alors la méthode se rapproche du template matching. A l'inverse, l'augmentation importante du nombre de candidats tend à se rapprocher de la méthode de déplacement intra-image (jusqu'au point où tous les candidats de l'image source se retrouvent dans la liste $\mathcal{L}_N$).

Au niveau de la complexité en terme de calcul, ce qui nécessite le plus d'opérations dans la méthode de *template matching* est la recherche dans l'image source du meilleur candidat. Celle-ci consiste à comparer un à un les pixels des *templates* par la méthode de SSD. Pour cela, tous les candidats potentiels de l'image source sont parcourus un à un. Dans ce cas, la complexité au décodeur de la méthode STMP est similaire à la prédiction par *template matching*.

Au codeur, il est nécessaire de tester tous les candidat de la liste $\mathcal{L}_N$ selon un critère débit-distorsion pour identifier le meilleur prédicteur de la liste, ce qui rend la méthode plus complexe. De même, l'intégration de la recherche au demi-pixel quadruple le nombre de candidats possibles, d'où une augmentation de la complexité d'un facteur similaire.

Les travaux présentés dans cette section ont donné lieu à la publication [99].

Troisième partie

Restauration d'image

CHAPITRE 5

Méthodes de régularisation d'image basées sur la variation totale : un état de l'art

Sommaire

La restauration est l'un des plus anciens problèmes posé en traitement d'image, elle demeure néanmoins très actuelle puisqu'elle a des répercussions dans de nombreux autres domaines. En effet, beaucoup d'applications, notamment en analyse d'image, nécessitent une étape de pré-traitement de l'image afin d'en améliorer la qualité. Une image se détériore pendant son acquisition, transmission et enregistrement. Il existe de nombreux types de dégradations : nous pouvons par exemple citer le flou engendrer par un mouvement pendant la prise de vue, le bruit venant de la transmission du signal, la détérioration dûe à l'algorithme d'encodage de l'image, etc. Le procédé consistant à supprimer ou diminuer les effets d'une telle détérioration s'appelle la *restauration*.

La méthodologie qui permet de répondre à ces problèmes de restauration repose sur une base mathématique solide, celle des équations aux dérivées partielles (EDP). En particulier, dans ce chapitre, cet outil est utilisé pour résoudre les problèmes mal posés exprimés sous forme variationnelle. Le formalisme mathématique des EDP définit un cadre unificateur qui permet d'adresser un grand nombre de problèmes posés en traitement d'image. Ce chapitre expose dans un premier temps le problème théorique. Il fournit ensuite les méthodes numériques les plus répandues permettant d'y répondre. Ce

chapitre n'a évidemment pas vocation à présenter toutes les variantes du modèle de régularisation disponibles dans la littérature ni tous les algorithmes pour résoudre ces problèmes. Nous verrons ensuite plusieurs exemples d'applications basées sur ce modèle théorique : débruitage, déconvolution, décomposition d'image en texture et structure, inpainting, reconstruction optimale d'une image codée et codage d'image. Toutes ces déclinaisons vont nous permettre de nous familiariser avec cet outil mathématique et serviront de base pour le chapitre de contribution suivant.

5.1 Problème théorique et méthodes numériques

5.1.1 Définition du problème

Le procédé de restauration de débruitage et défloutage, tel qu'il est défini dans un grand nombre d'articles (voir par exemple [120]), consiste à retrouver une image u à partir d'une observation g donnée par :

$$g = Hu + b \tag{5.1}$$

où b caractérise le bruit généralement considéré comme gaussien de moyenne nulle et de variance σ, et H un opérateur linéaire de convolution. Pour retrouver u à partir de g, la première idée consiste à inverser le problème par $u = H^{-1}g$. Or, la matrice H peut-être mal conditionnée ou non inversible, ce qui conduit à un grand nombre de solutions ; nous parlons alors de problème inverse mal posé. De ce fait, la régularisation du problème cherche une approximation de u vérifiant certaines hypothèses de régularité. La prochaine section est dédiée à la résolution de ce problème, d'abord par la méthode de Tikhonov et ensuite par une méthode basée sur la norme de la variation totale (TV). En nous appuyant sur cette dernière, nous déclinerons enfin plusieurs modèles de régularisation selon le type d'application souhaité.

5.1.2 Régularisation d'un problème inverse mal posé

Le problème de la déconvolution en présence de bruit est mal posé au sens d'Hadamard. En d'autres termes, cela signifie que la solution ne répond pas aux trois critères qui sont l'existence, l'unicité et la stabilité de la solution (une faible perturbation des données conduisant à une faible perturbation de la solution). Il est alors nécessaire de régulariser la solution par l'introduction de contraintes *a priori*. Lorsque le noyau de convolution et les statistiques du bruit sont connus, calculer l'estimateur du maximum de vraisemblance consiste à minimiser une énergie exprimant la contrainte de régularisation. Si le bruit additif b est supposé gaussien, la méthode du Maximum de Vraisemblance nous conduit à chercher u comme solution du problème de minimisation d'une fonctionnelle $J(u)$:

$$\inf_{u(\mathbf{x})} \left\{ J\left(u(\mathbf{x})\right) = \|g(\mathbf{x}) - Hu(\mathbf{x})\|_2^2 \right\} \tag{5.2}$$

avec $\mathbf{x} = (x_1, x_2)$ le vecteur des variables et où $\|\cdot\|_2$ représente la norme L^2, soit $\|g(\mathbf{x}) - Hu(\mathbf{x})\|_2^2 = \int_\Omega \left(g(\mathbf{x}) - Hu(\mathbf{x})\right)^2 d\Omega$ avec $\Omega \subset \mathbb{R}^2$ le support de l'image. J est appelée fonctionnelle car elle prend en argument une fonction et peut s'écrire sous la forme d'une intégrale. La notation $\int_\Omega u(\mathbf{x}) d\Omega$

représente une intégrale sur le domaine de Ω. Dans le cas d'une image plane, nous avons alors $\int_\Omega u(\mathbf{x})d\Omega = \int_{x_1}\int_{x_2} u(x_1,x_2)dx_1dx_2$.

Pour simplifier la notation, nous introduisons u comme étant $u(\mathbf{x})$. Selon cette notation, la solution $\hat{u}$ s'exprime par :

$$\hat{u} = \arg\min_u J(u) \tag{5.3}$$

5.1.2.1 Régularisation de Tikhonov

L'image est supposée *a priori* lisse, ce qui nous permet d'introduire un terme de régularisation qui impose une contrainte de "continuité" sur la solution à restaurer [141] :

$$\inf_u \left\{ J(u) = \frac{\lambda}{2}\left\|g - Hu\right\|_2^2 + \left\|\nabla u\right\|_2^2 \right\},\ \lambda > 0 \tag{5.4}$$

où le premier terme, $\|g - Hu\|_2^2$, est appelé le terme d'attache aux données (ou terme de fidélité) et le second, $\|\nabla u\|_2^2$, le terme de régularisation. Le paramètre λ est un coefficient de régularisation et le champ de vecteurs ∇ exprime le gradient, soit :

$$\nabla u(\mathbf{x}) = \overrightarrow{grad}\, u(\mathbf{x}) = \left(\frac{\partial u}{\partial_{x_1}}, \frac{\partial u}{\partial_{x_2}}\right) \tag{5.5}$$

Le gradient indique la direction et l'intensité de la plus grande variation du champ scalaire u.

Le paramètre lagrangien λ pondère l'importance accordée à la minimisation du terme de fidélité. Quand λ est proche de zéro, le terme de régularisation devient prépondérant, ce qui peut donner comme solution une image nulle (donc très régulière). Dans le cas inverse, la minimisation de la fonctionnelle J revient à chercher une solution aux moindres carrés. En d'autres termes, nous cherchons u tel que Hu soit proche de g et que u soit régulier.

La résolution de cette minimisation en considérant la norme L^2 utilisée par Tikhonov [141] ne fournit pas toujours un résultat acceptable, car la fonction quadratique a tendance à sur-régulariser le problème en fournissant des images floues.

5.1.2.2 Régularisation basée sur la variation totale

La régularisation de Tikhonov étant trop "violente", d'autres alternatives ont été explorées. L'idée est de remplacer le terme de régularisation $\|\nabla u\|_2^2$ par une norme moins pénalisante pour les gradients élevés. Rudin, Osher et Fatemi [121] ont proposé un modèle de régularisation basé sur la variation totale (TV [1]) qui permet d'effectuer une diffusion non linéaire. La TV mesure la somme des variations d'un signal.

La variation totale est une norme L^1 calculant la valeur absolue de la magnitude du gradient (différence du premier ordre). Il est montré dans [122] que la norme TV est plus approprié aux images naturelles que la norme L^2, car elle supprime les oscillations tout en conservant les contours de

1. *Total Variation*

l'image. Cela conduit à une minimisation de fonctionnelle dans un espace de Banach particulier, mais bien adapté au problème : l'espace des fonctions à variation bornée (espace BV, voir [15]). Ainsi :

$$BV(\Omega) = \left\{ f \in L^1(\Omega) \mid J(f) < +\infty \right\} \tag{5.6}$$

Cet espace est d'une grande utilité dans le cadre du traitement des images car il autorise la présence de discontinuités, autrement dit de contours. Ainsi les images obtenues par cette minimisation présentent des bords francs. La variation totale d'une fonction f est ainsi définie par :

$$\|f\|_{TV} = \int_{-\infty}^{+\infty} \left| f'(t) \right| dt \tag{5.7}$$

Dans le cas 2-D, la notation $\nabla f\,(x, y)$ est privilégiée, où $\nabla f\,(x, y) = \frac{\partial}{\partial x} \frac{\partial f}{\partial y}$.

La fonction f est à variation totale bornée si :

$$\|f\|_{TV} < +\infty$$

L'existence, l'unicité et la stabilité de la solution au problème de régularisation basée sur la norme TV est débattue par Chambolle et Lions dans [35].

En reprenant l'équation 5.4 avec cette norme, la régularisation d'une image basée sur la variation totale consiste à minimiser $J(u)$ tel que :

$$\min \left\{ J(u) = \frac{\lambda}{2} \|g - Hu\|_2^2 + \|u\|_{TV} \right\}, \lambda > 0 \tag{5.8}$$

L'étude mathématique de 5.8 a été faite dans Combettes et Pesquet dans [49].

Les méthodes de Tikhonov et ROF (Rudin-Osher-Fatemi) sont comparées sur la figure 5.1 qui illustre la régularisation d'une image bruitée (en référence à la section 5.2.1). Alors que le terme de régularisation dans la méthode de Tikhonov estompe les contours, l'approche basée sur la variation totale débruite l'image en gardant l'aspect net des contours.

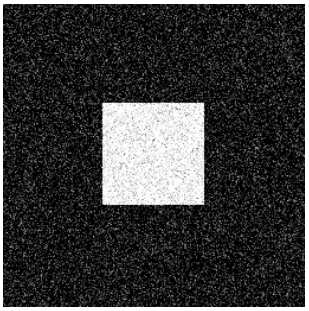
5.1.2.3 Généralisation

Dans le but de généraliser le critère de régularisation, nous utiliserons la formulation Φ [27] qui exprime la fonctionnelle par :

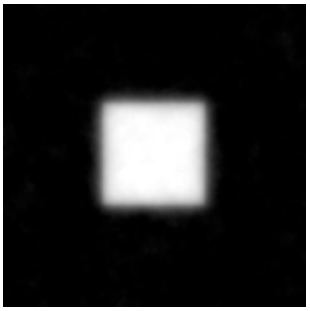
$$J(u) = \frac{\lambda}{2} \int_\Omega (g(\mathbf{x}) - Hu(\mathbf{x}))^2 d\Omega + \int_\Omega \Phi(|\nabla u(\mathbf{x})|) d\Omega \tag{5.9}$$

Pour $\Phi(|\nabla u(\mathbf{x})|) = |\nabla u(\mathbf{x})|$, la fonction $\Phi(\cdot)$ correspond à la variation totale. Une multitude de fonctions différentes existent [111, 64, 46, 14, 65, 142].

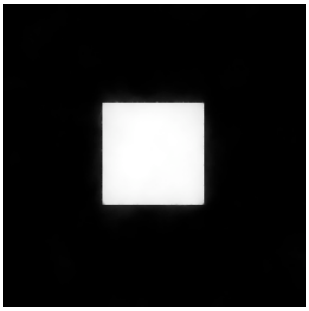
Les équations d'Euler-Lagrange associées, pouvant être résolues par une descente de gradient, s'expriment sous la forme ([52]) :



(a) image bruitée



(b) régularisation de Tikhonov



(c) régularisation de Rudin et al.

FIGURE 5.1 – Régularisation d'une image bruitée (a) par la méthode de Tikhonov (b) [141] et de Rudin et al. (c) [121].

$$\lambda \left[H^*(g - Hu)\right] + div \left(\Phi' \left(|\nabla u| \right) \frac{\nabla u}{|\nabla u|} \right) = 0 \tag{5.10}$$

avec les conditions aux limites dites de Neumann et où H^* est l'adjoint de H (transposée de la matrice conjuguée de H). L'opérateur div exprime la divergence tel que ([155]) :

$$div\ \mathbf{A} = \nabla \cdot \mathbf{A} = \frac{\partial A_1}{\partial x_1} + \frac{\partial A_2}{\partial x_2} \tag{5.11}$$

Ainsi, lorsque la fonction $\Phi(\cdot)$ exprime la variation totale, le terme de régularisation donné par $\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right)$ (Eq. 5.10) correspond à la divergence du gradient normalisé, soit la courbure moyenne des lignes de niveau de l'image u [39]. Le paragraphe 5.1.3.2 est dédié à l'expression de la courbure dans un espace discret.

La réécriture de 5.10 permet d'en comprendre le sens géométrique :

$$\lambda \left[H^*(g - Hu)\right] + \Phi'' \left(|\nabla u| \right) u_{\xi\xi} + \frac{\Phi' \left(|\nabla u| \right)}{|\nabla u|} u_{\eta\eta} = 0 \tag{5.12}$$

où $u_{\xi\xi}$ et $u_{\eta\eta}$ représentent respectivement la dérivée seconde de u dans la direction du gradient et la direction orthogonale au gradient. Ainsi, le processus de diffusion anisotrope opère dans deux directions orthogonales pondérées différemment ($\Phi'' \left(|\nabla u| \right)$ et $\frac{\Phi'(|\nabla u|)}{|\nabla u|}$). Le cas particulier d'une régularisation isotrope intervient lorsque les deux paramètres pondérants sont identiques, correspondant à une fonction Φ quadratique (soit le modèle de régularisation de Tikhonov).

Le modèle a été étendu aux images multi-canaux. L'idée sous-jacente est de "mixer" les trois composantes couleur afin d'améliorer le résultat. Le sujet a été traité entre autres dans [124, 28].

5.1.3 Méthodes numériques

5.1.3.1 Résolution par un algorithme de descente de gradient

L'équation d'Euler-Lagrange 5.10 n'a pas de solution simple, la principale difficulté venant de la linéarisation du terme non-linéaire de divergence. Dans [121], Rudin et al. proposent une méthode de résolution basée sur un algorithme de descente de gradient . Dans ce cadre, l'image est considérée comme une fonction d'espace et de temps, pour laquelle nous recherchons l'état stationnaire donné par l'équation d'évolution :

$$\begin{aligned} \frac{\partial u}{\partial t} &= \lambda \left[H^*(g - Hu)\right] + div(\Phi'\left(|\nabla u|\right) \frac{\nabla u}{|\nabla u|}) \\ &= \nabla J(u) \end{aligned} \tag{5.13}$$

Afin d'éviter la division par zéro, $|\nabla u|$ est généralement remplacé par $\sqrt{|\nabla u|^2 + \epsilon}$, avec ϵ petit ([149]). Dans l'implémentation numérique, la mise à jour de l'image à chaque itération est effectuée par :

$$u^{(n+1)} = u^{(n)} - \gamma_n \nabla J\left(u^{(n)}\right) \tag{5.14}$$

Ceci garantit à l'algorithme de converger vers une solution, et où γ est le paramètre du pas de l'algorithme de descente de gradient, habituellement fonction du nombre d'itérations effectuées n. Enfin, $u^{(0)}$ est initialisé par $u^{(0)} = g$.

La méthode de Rudin et al. [121] peut converger très lentement à cause de ces contraintes de stabilité. Plusieurs auteurs ont proposé une amélioration de la méthode. Dans [149], Vogel et Oman ont proposé un algorithme basé sur un schéma d'itération de méthode de point fixe qui permet de résoudre directement l'équation d'Euler-Lagrange. En prenant la norme TV, la méthode consiste à résoudre $u^{(n+1)}$ (Eq. 5.10) :

$$\lambda \left[H^*(g - Hu^{(n+1)})\right] + div\left(\frac{\nabla u^{(n+1)}}{|\nabla u^n|}\right) = 0 \tag{5.15}$$

La méthode est robuste mais converge de façon linéaire. D'autres approches basées sur la méthode de Newton sont présentées par Chan et al. dans [37] et Chambolle dans [34]. Egalement, une méthode rapide est présentée par Darbon et Sigelle [51] pour notamment reformuler le problème en termes de champs de Markov binaire, et ainsi de le résoudre à l'aide de graph-cuts. Enfin, la méthode "split Bregman" [66] est à ce jour l'une des plus efficace pour optimiser la régularisation basée sur la norme L_1.

5.1.3.2 Courbure d'un espace discret

L'expression de la courbure $\nabla \cdot \left(\frac{\nabla u}{|\nabla u|}\right)$ dans la formulation du problème de régularisation basée sur la variation totale nous amène à définir une approximation discrète de cette formule. Pour illustrer la méthode de calcul, prenons un pixel au point O et ses huit voisins (Fig. 5.2). Les points n, e, s et w sont des points intermédiaires. Soit $\mathbf{v} = (v_1, v_2) = \frac{\nabla_{\mathbf{x}} u_O}{|\nabla_{\mathbf{x}} u_O|}$.

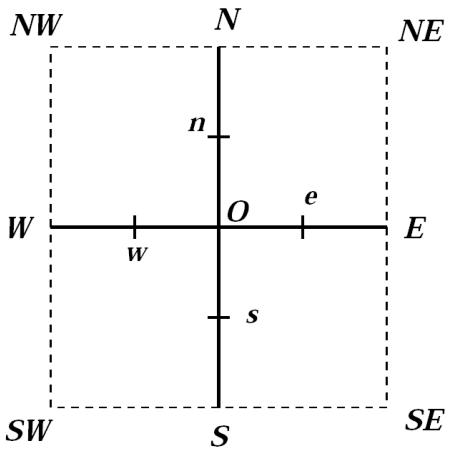


FIGURE 5.2 – Calcul de la courbure dans un espace discret : un pixel au point O et ses huit voisins. (source : [39])

D'après la définition de la divergence Eq. 5.11 :

$$\nabla_{\mathbf{x}} \cdot \mathbf{v} = \frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} \tag{5.16}$$

Dans un espace discret, nous obtenons l'approximation suivante :

$$\nabla_{\mathbf{x}} \cdot \mathbf{v} \approx \mathbf{v}_e - \mathbf{v}_w + \mathbf{v}_n - \mathbf{v}_s \tag{5.17}$$

avec $\mathbf{v}_e$ l'expression du gradient normalisé au point e.

Une des interprétations possibles est qu'au point O sera calculée la somme des variations des gradients normalisés au voisinage de O selon l'axe des lignes et des colonnes. L'objectif est de minimiser cette variation afin d'obtenir un rendu lisse.

5.2 Exemples d'applications

Nous allons illustrer le modèle de régularisation basé sur la minimisation de la variation totale vue précédemment dans différents contextes d'utilisation. Selon les applications, le modèle est soumis à des ensembles de contraintes dépendant de la nature de la restauration à réaliser. La déclinaison de ces modèles est présentée dans les parties suivantes.

5.2.1 Débruitage

Le débruitage est un cas particulier de l'équation 5.9 où nous cherchons à obtenir une image u plus régulière que g. Fatemi, Rudin et Osher ont été les premiers, en 1992, à concevoir un algorithme efficace de mise en oeuvre de ce concept exposé [121]. Nous considérons ici le cas d'une image bruitée mais sans flou ; H correspond alors à une matrice identité I, donc $Hu = u$. La fonctionnelle J est réécrite selon :

$$J(u) = \frac{\lambda}{2} \int_{\Omega} (g(\mathbf{x}) - u(\mathbf{x}))^2 d\Omega + \int_{\Omega} \Phi(|\nabla u(\mathbf{x})|) d\Omega \tag{5.18}$$

En considérant la semi-norme TV, la fonction d'évolution donnée en Eq. 5.13 se réécrit :

$$\nabla J(u) = \lambda(g - u) + div\left(\frac{\nabla u}{|\nabla u|}\right) \tag{5.19}$$

La figure 5.3 illustre les résultats obtenus sur une image dégradée avec un bruit gaussien additif de variance $\sigma_b = 20$. Nous procédons à 100 itérations de l'algorithme de descente de gradient pour obtenir une solution.

Cependant, certains détails fins sont supprimés au cours cette opération de débruitage du fait de leurs oscillations rapides autour d'une moyenne locale. Ce constat est très marquant pour les textures rigides fines, où le débruitage tend à remplacer la texture par une valeur constante. Pour remédier à cela, Gilboa et al. [65] ont proposé une nouvelle contrainte au modèle ROF dont le but est d'adapter localement le terme de fidélité en fonction de la variance locale de la partie oscillante du signal. Les résultats décrits dans l'article montrent l'amélioration de la méthode pour certains exemples d'image.

5.2.2 Déconvolution

Le flou qui apparaît sur certaines photographies peut par exemple être dû aux mouvements des objets ou aux erreurs de calibration des appareils de capture. En règle générale, H est considéré comme un opérateur linéaire de convolution avec un noyau symétrique positif k, la plupart du temps gaussien, avec $Hu = k \star u$. Mais dans la plupart des applications, il est rare de connaître exactement le type et l'intensité du flou d'une observation g. La difficulté de l'approche réside donc dans le manque de connaissance sur k ; le traitement de restauration est alors appelée déconvolution aveugle. L'exercice consiste à retrouver u et k seulement à partir de l'image observée g. Afin d'y parvenir, plusieurs approches ont été élaborées : filtrage inverse [81, 105], méthodes statistiques [86, 79], et méthodes de régularisation [42, 131, 161]. Nous nous intéresserons particulièrement aux méthodes de régularisation basées sur la variation totale. La méthode présentée dans [42] définit le problème de déconvolution par :

$$\inf_{u,k} \left\{ J(u,k) = \frac{1}{2} \int_{\Omega} (g - k \star u)^2 d\Omega + \lambda_1 \int_{\Omega} |\nabla u| \, d\Omega + \lambda_2 \int_{\Omega} |\nabla k| \, d\Omega \right\} \tag{5.20}$$

où λ_1 et λ_2 sont des paramètres positifs qui définissent le degré de régularité respectif à u et k. Dans ce cas, la fonctionnelle J dépend à la fois de u et k, ce qui nous conduit à résoudre un problème de minimisation conjointe. La motivation qui a conduit les auteurs à utiliser la norme TV pour k vient du fait que k peut, à l'instar de u, contenir des contours (les quatre types récurrents de noyau k sont présentés dans [83]). Les équations d'Euler-Lagrange associées à 5.20 nous donnent :

$$\frac{\partial J}{\partial k} = \hat{u} \star (g - k \star u) + \lambda_2 \nabla \cdot \left(\frac{\nabla k}{|\nabla k|}\right) = 0 \tag{5.21}$$

$$\frac{\partial J}{\partial u} = \hat{k} \star (g - k \star u) + \lambda_1 \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) = 0 \tag{5.22}$$

où $\hat{h}$ et $\hat{u}$ sont les L^2-conjugués de k et u respectivement. Pour résoudre ces deux équations, un algorithme de minimisation procède en alternance pour résoudre k^{n+1} et u^{n+1} en assurant à $J(u^n, k^n)$ de toujours décroître quand n croît. Plusieurs contraintes sont imposées à k pour respecter les connaissances *a priori*, à savoir : $\int_\Omega k \, d\Omega = 1$, $k(\mathbf{x}) \geq 0$ et $k(\mathbf{x}) = k(-\mathbf{x})$ (centrosymétrie). Tandis que l'image u est initialisée de façon classique par $u^0 = g$, la fonction k est initialisée par la fonction de dirac $\delta(\mathbf{x})$ (solution de k lorsqu'il n'y a pas de flou), avec $\delta \star u = u$. La méthode proposée dans [42] converge rapidement tout en permettant de retrouver les bonnes caractéristiques de l'image, cependant le fait de résoudre successivement deux systèmes d'équation pour u et k peut engendrer une complexité calculatoire non négligeable.

La littérature propose plusieurs optimisations de l'algorithme de minimisation de u et k. Notamment, une méthode très efficace basée sur les itérations de Bregman est présentée dans [72].

5.2.3 Décomposition d'une image en structure/texture

La minimisation d'une fonctionnelle basée sur la variation totale a également été utilisée pour extraire différentes composantes d'une image. Le but est de décomposer une image dans différents espaces fonctionnels afin de mieux discerner le fond structurel (aspect géométrique, aussi appelé image "cartoon") du remplissage des régions (aspect texturel). Il existe plusieurs modèles de décomposition en structure/texture dans la littérature basés sur une mesure de l'activité locale. Le concept est de séparer l'image f en deux composantes u et v telle que u représente la structure de l'image (contours + fond) et que v soit la partie oscillante de l'image (+ éventuellement le bruit).

Pour extraire la structure u dans l'espace BV et la texture/bruit v comme une fonction d'oscillation, Meyer proposa le modèle suivant [98] :

$$\inf_{u \in BV} \left\{ J(u) = \int |\nabla u| + \lambda \left\| v \right\|_G , \; v = f - u \right\} \tag{5.23}$$

où G désigne l'espace de Banach [98] composé de toutes les fonctions généralisées $v(\mathbf{x})$ dans $\mathbb{R}^n$ pouvant s'écrire sous la forme :

$$v = div\,(\vec{g}) \,, \; \vec{g} = [g_i]_{i=1,\ldots,n} \in L^\infty\left(\mathbb{R}^n; \mathbb{R}^n\right) \tag{5.24}$$

Sa norme $\left\| v \right\|_G$ est définie comme la borne inférieure de toutes les normes L^∞ des fonctions $|g(\mathbf{x})|$ sur toutes les décompositions de f (5.24). En un sens, G peut être interprété comme l'espace dual de BV. Cependant, il n'est pas possible d'écrire les équations d'Euler-Lagrange associées à 5.23, et ainsi de pouvoir recourir aux équations à dérivées partielles pour résoudre ce problème. Plusieurs modèles ont été proposés pour approximer Eq.5.23. C'est le cas du modèle de Vese-Osher [145] (étendu aux images couleurs RGB dans [146]) :

$$\inf_{u \in BV,\, \vec{g} \in C_0^1(\mathbb{R}^2; \mathbb{R}^2)} \left\{ J_p(u, \vec{g}) = \int |\nabla u| + \lambda \int |f - u - div\,(\vec{g})|^2 + \mu \left[\int |\vec{g}|^p \right]^{\frac{1}{p}} \right\}, \text{ avec } p \geq 1 \tag{5.25}$$

Dans [145], les auteurs ont montré que les résultats pour $1 \leq p \leq 10$ sont très similaires. Pour $p = 1$, la minimisation de J en fonction de u et $\vec{g} = (g_1, g_2)$ donne les équations d'Euler-Lagrange suivantes :

$$\begin{array}{rcl} u & = & f - \partial_1 g_1 - \partial_2 g_2 + \frac{1}{2\lambda} div\left(\frac{\nabla u}{|\nabla u|}\right) \\ \mu \frac{g_1}{\sqrt{g_1^2+g_2^2}} & = & 2\lambda\left[\partial_1(u-f) + \partial_{11}^2 g_1 + \partial_{12}^2 g_2\right] \\ \mu \frac{g_2}{\sqrt{g_1^2+g_2^2}} & = & 2\lambda\left[\partial_2(u-f) + \partial_{12}^2 g_1 + \partial_{22}^2 g_2\right] \end{array} \tag{5.26}$$

Le modèle de [145] est illustré sur la figure 5.4.

Un autre modèle, appelé $TV - L^1$ [159, 36], remplace la norme L^2 du terme d'attache aux données $f - u$ du modèle de Meyer (Eq. 5.23) par la norme L^1. Pour une étude comparative entre le modèle de Meyer et le modèle $TV - L^1$, se référer à [160].

Bien que différent du problème de restauration d'image vu jusqu'ici, ce type de méthode ouvre la voie à de nombreuses applications où l'intérêt est d'utiliser une approche adéquate et différente pour chacune des deux composantes u et v. Par exemple, un algorithme de recouvrement d'une partie manquante peut dans ce cas s'inspirer d'une méthode de synthèse de texture par *patch* pour la partie oscillante v et d'une méthode par diffusion d'EDP pour propager les contours de u.

5.2.4 Inpainting

Plusieurs contributions sont présentées dans cette section. Nous porterons notre attention sur deux catégories différentes d'applications. La première concerne la restauration de pixels manquants dans une image. La deuxième se focalise sur la restauration de coefficients en ondelettes perdus lors d'une transmission. Bien que l'approche soit dissemblable, la localisation de l'information manquante est considérée connue dans les deux cas.

5.2.4.1 Domaine spatial

L'inpainting dans le domaine direct est l'opération qui consiste à restaurer des images dont certaines parties ou groupes de pixels sont manquants. Le procédé d'inpainting est décrit par Bertalmio et al. dans [23] comme suit :

> *"The modification of images in a way that is non-detectable for an observer who does not know the original image is a practice as old as artistic creation itself. [...] This practice is called retouching or inpainting. The objective of inpainting is to reconstitute the missing portions of the work, in order to make it more legible and to restore its unity."*

Les algorithmes d'inpainting sont employés dans diverses applications ; nous pouvons citer par exemple la restauration numérique de peintures anciennes à des fins de conservation, la restauration de photographies ou films fortement dégradés (rayures, parties manquantes), la désoccultation [96, 97] (capacité à retrouver des régions occultées d'une image par interpolation avec son voisinage), le zoom numérique [69], le codage vidéo [40], l'amélioration d'image codées avec perte [142], la suppression de texte ou de logo en surimpression d'une séquence [39, 144, 143], etc. Ces applications étant souvent très proches, les articles de la littérature décrivent la plupart du temps un modèle

qui s'applique dans différents types d'utilisation. Dans cette section, nous traiterons le problème particulier de désoccultation (ou *local inpainting*) par une approche variationnelle. Certaines autres applications sont étudiées par la suite.

Pour reprendre le formalisme introduit dans 5.10, H est ici un opérateur de masquage qui supprime certaines parties de l'image. Soit le domaine $D \in \Omega$ tel que :

$$(Hu)(\mathbf{x}) = \begin{cases} 0 & \text{si} \quad \mathbf{x} \in D \\ u(\mathbf{x}) & \text{si} \quad \mathbf{x} \notin D \end{cases} \tag{5.27}$$

Nous appelons E le domaine correspondant à l'extension de D dans l'image Ω, avec $E \subset \Omega$. Dans [40], T.F. Chan et al. évoquent trois grands principes auxquels le modèle d'inpainting doit obéir :

1. Le modèle doit résoudre le problème localement. En effet, les modèles d'inpainting par minimisation d'une fonctionnelle ne reposent pas sur un apprentissage global de l'image, de ce fait u_D (la région à restaurer du domaine D) est complètement déterminée à partir du voisinage de D dans g.
2. Le modèle doit être capable de restaurer les contours disjoints. Ce critère est important pour la compréhension de la scène, que ce soit pour un observateur humain ou une méthode de reconnaissance d'objet ou de segmentation d'image.
3. Le modèle doit être robuste au bruit : la vision humaine est douée pour extraire les caractéristiques d'une image bruitée et étendre l'information au domaine D.

Le modèle d'inpainting [38] consiste à minimiser la fonctionnelle J selon :

$$J(u) = \frac{\lambda}{2} \int_E (g(\mathbf{x}) - u(\mathbf{x}))^2 d\Omega + \int_{E \cup D} \Phi(|\nabla u(\mathbf{x})|) d\Omega \tag{5.28}$$

Ainsi, l'image doit être la plus ressemblante à l'observation aux abords de D grâce au terme de fidélité. Le terme de régularisation opérant dans le domaine $E \cup D$ interpole l'image. L'équation 5.28 peut être simplifiée par $J(u) = \int_{E \cup D} \Phi(|\nabla u(\mathbf{x})|) d\Omega$ si l'image est considérée comme non bruitée.

L'équation de descente de gradient pour J est donnée par :

$$\frac{\partial u}{\partial t} = \lambda_e \, (g - u) + div \left(\Phi' \left(|\nabla u| \right) \frac{\nabla u}{|\nabla u|} \right) \tag{5.29}$$

où λ_e est le multiplicateur de Lagrange définit par :

$$\lambda_e = \begin{cases} \lambda, & \mathbf{x} \in E \\ 0, & \mathbf{x} \in D \end{cases} \tag{5.30}$$

Ainsi, le modèle d'inpainting débruite l'image aux abords de la zone à restaurer (pour $\mathbf{x} \in E$) grâce au terme d'attache aux données (identique au modèle de débruitage présenté en 5.2.1). Ce mécanisme rend la méthode plus robuste au bruit.

La figure 5.3 illustre quelques résultats d'inpainting en fonction du nombre de pixels supprimés.

Dans [24], Bertalmio et al. proposent une approche différente basée au préalable sur une décomposition en structure/texture de l'image à restaurer selon le modèle de Vese-Osher [145] présenté en 5.2.3 (le même principe est repris plus tard par Elad et al. dans [58]). L'hypothèse de départ,

formulée d'après [116], montre que le résultat d'une méthode d'inpainting dépend ostensiblement du contenu de l'image à traiter. Dans une image partitionnée en blocs, le modèle de [116] va adapter la méthode d'inpainting selon que le bloc soit qualifié de structure ou de texture. Partant de ce principe, Bertalmio et al. [24] vont utiliser une méthode d'inpainting différente selon la composante de structure ou de texture. Pour la composante de texture, la méthode utilise l'algorithme de synthèse de texture de Efros et Leung [57] présenté en 3.3.2.1. Pour restaurer la partie manquante dans la composante structure, Bertalmio et al. s'appuient sur leurs travaux antérieurs [23] basés sur la diffusion d'isophotes. Au final, l'image est reconstruite par addition des deux composantes de structure et texture.

Nous pouvons faire le parallèle avec la méthode d'inpainting proposée par Yamauchi dans [157], où l'idée principale consiste à combiner les méthodes de synthèse de texture et d'inpainting sur une représentation adéquate de l'image. Dans un premier temps, l'image est décomposée selon une partie haute-fréquence et une partie basse-fréquence obtenues par la sélection d'un sous-ensemble des coefficients DCT de l'image. La partie D manquante de la composante basse-fréquence est restaurée à l'aide de l'algorithme de Oliveira et al. [108] qui a la particularité d'être très rapide puisqu'il repose sur une diffusion isotropique réalisée grâce à un simple filtrage. Bien que ce procédé tend à lisser les contours, l'utilisation de ce type de méthode sur une composante basse-fréquence d'une image a du sens. Quant à la partie haute-fréquence de l'image, Yamauchi et al. proposent d'utiliser la méthode de Wei et Levoy [152] (voir le chapitre 3).

Le modèle de régularisation présenté en 5.28 peut également être appliqué au problème de sur-échantillonnage des images ; nous parlons dans ce cas de super-résolution ou zoom numérique. L'interpolation est alors considérée comme un problème de restauration d'une image dont les pixels connus sont équitablement répartis. Les méthodes sont notamment décrites par dans Tschumperlé dans[143] et Guichard et al. dans[69].

5.2.4.2 Domaine ondelette

Un contexte d'utilisation très intéressant a été décrit par T.F. Chan et al. dans [41]. Bien que l'approche soit fortement liée aux applications précédentes, elle diffère cependant par le fait que la partie manquante ou endommagée est représentée dans le domaine ondelettes. Nous nous plaçons notamment dans le cas d'utilisation où une partie des coefficients d'ondelettes sont perdus pendant la transmission d'une image JPEG2000. De ce fait, la nature du problème d'inpainting s'en trouve changée, car un coefficient ondelette supprimé peut entraîner l'altération de plusieurs pixels dans le domaine spatial. La région à restaurer n'étant pas clairement définie dans le domaine spatial, l'utilisation de modèle d'inpainting de T.F. Chan [38] vu en 5.2.4.1 ne peut pas s'appliquer ici. A l'inverse, l'interpolation directe dans le domaine ondelette n'est pas non plus envisageable puisque les coefficients ondelettes sont calculés pour décorréler le signal. De plus, contrairement aux problèmes de débruitage (voir 5.2.1) pour lesquels les dégradations sont homogènes sur toute l'image, le problème d'inpainting dans le domaine ondelette est souvent non-homogène, c'est-à-dire qu'une région de l'image est plus touchée par les dégradations qu'une autre.

Il semble nécessaire, dans cet exercice d'inpainting des coefficients d'ondelettes, de contrôler la régularité dans le domaine spatial pour conserver les caractéristiques géométriques importantes de l'image. A cette fin, les auteurs de [41] se sont inspirés des méthodes de régularisation par minimisation de la variation totale, mais cette fois-ci appliquées dans le domaine ondelette. Cette combinaison entre la TV dans le domaine pixellique et les coefficients d'ondelettes a préalablement

été explorée dans par les mêmes auteurs [44] dans un contexte de débruitage et de compression d'image.

Le modèle présenté dans [41] consiste à restaurer les coefficients ondelettes $\beta_{j,k}$, avec $j \in \mathbb{Z}$ et $k \in \mathbb{Z}^2$, pour $(j,k) \in I$ le domaine des coefficients dégradés :

$$\min_{\beta_{j,k}:(j,k)\in I} J(u) = \int_{\mathbb{R}^2} |\nabla_x u(\beta, \mathbf{x})| \, d\mathbf{x} \tag{5.31}$$

Nous retrouvons ici le terme de régularisation classique du modèle ROF sans le terme d'attache aux données. L'image u est définie en fonction des coefficients ondelettes β par :

$$u\,(\beta, \mathbf{x}) = \sum_{j,k} \beta_{j,k} \psi_{j,k}(\mathbf{x}) \tag{5.32}$$

où ψ est la base d'ondelettes utilisée pour transformer l'image. La contrainte appliquée dans le domaine ondelette s'exprime par :

$$\beta_{j,k} = \alpha_{j,k}, \;\; (j,k) \notin I \tag{5.33}$$

où α représente les coefficients ondelettes non dégradés de l'image d'origine u_0. Cette contrainte assure que seuls les coefficients dégradés de I vont être restaurés. L'équation d'Euler-Lagrange associée à 5.31 donne (voir [41] pour le détail des calculs) :

$$-\int_{\mathbb{R}^2} \nabla \cdot \left[\frac{\nabla u}{|\nabla u|} \right] \psi_{j,k} d\mathbf{x} = 0 \tag{5.34}$$

Cette équation fait le lien entre la courbure de l'image et le domaine ondelette. L'article [41] présente un algorithme de descente de gradient classique pour résoudre le problème de minimisation de l'Eq. 5.31 à partir des équations d'Euler-Lagrange de l'Eq. 5.34.

5.2.5 Amélioration des images compressées avec perte

Les dégradations visuelles à bas-débit des standards de codage d'image sont bien connues, que ce soit pour un codage basé DCT (type JPEG [150]) ou ondelettes (type JPEG2000 [139]). En particulier, le codage par transformée DCT souffre d'un "effet de bloc" [76], dû au partitionnement de l'image, et d'artefacts d'oscillations autour des contours "*ringing artifact*", connu sous le nom de phénomène de Gibbs et dû au procédé destructeur de quantification des coefficients transformés.

Pour gommer ces artefacts, une méthode simple consiste à considérer ces dégradations comme un bruit additif à l'image d'origine. La technique générique de régularisation d'image vue en 5.2.1 peut alors être employée pour améliorer l'image décodée (exemples dans [142]). Le résultat donnera une image plus lisse avec des contours marqués grâce à la norme TV utilisée. Lorsque les artefacts sont très marqués, la méthode peut cependant se révéler contre-productive.

Cependant, il existe une famille d'algorithmes par "reconstruction optimale" plus efficaces pour résoudre ce type de problème. Dans ce cas, nous considérons que les dégradations dûes au codage de l'image découlent de la quantification des coefficients transformés. En effet, au cours du décodage,

Algorithme 5.1 Méthode générique de reconstruction optimale d'images compressées avec perte par minimisation contrainte de la variation totale.

1. Soit $u^{(0)}$ l'image décodée et $n = 0$ l'itération courante
2. Soit K le nombre maximum d'itérations et ϵ la différence minimale entre deux images successives
3. Tant que $n < K$ et $E(u^{(n)}, u^{(n-1)}) > \epsilon$, faire
4. Calculer $v^{(n)} = u^{(n)} - \gamma_n \nabla J\left(u^{(n)}\right)$, avec $J(u) = \int_\Omega |\nabla u| \, d\Omega$ (voir équation 5.14)
5. Mettre tous les coefficients $\left(T\left(v^{(n)}\right)\right)_{i,j} < Q^-_{i,j}$ à $Q^-_{i,j}$
6. Mettre tous les coefficients $\left(T\left(v^{(n)}\right)\right)_{i,j} > Q^+_{i,j}$ à $Q^+_{i,j}$
7. Calculer $u^{(n+1)} = T^{-1}\left(v^{(n)}\right)$ et $n = n + 1$
8. Fin de la boucle, l'image décodée est donnée par $u^{(n)}$

un coefficient $c_{i,j}$ à la position (i, j) d'un bloc de l'image subit une quantification inverse le plaçant à une valeur fixe comprise dans son intervalle de quantification $\overline{c_{i,j}} \in [Q^-_{i,j}, Q^+_{i,j}]$. En théorie, n'importe quelle valeur de l'intervalle de quantification peut être la bonne, c'est-à-dire la valeur correspondante au coefficient d'origine. Le problème est schématisé sur la figure 5.6 qui illustre une fonction classique de quantification scalaire uniforme appliquée aux coefficients transformés. Les méthodes de décodage par reconstruction optimale qui nous intéressent reposent sur l'idée que l'image décodée doit minimiser la variation totale de l'image en faisant osciller les coefficients dans leurs intervalles de quantification respectifs. En d'autres termes, le procédé de régularisation d'image par minimisation de la variation totale est soumis à une contrainte supplémentaire appliquée aux coefficients transformés.

L'algorithme général, inspiré de [12, 13, 164], est décrit dans 5.1, avec $T(\cdot)$ et $T^{-1}(\cdot)$ les fonctions de transformation et transformation inverse de l'image (ici la DCT) et $E(\cdot, \cdot)$ la fonction de distance entre deux images. Nous pouvons remarquer que dans l'algorithme, le terme d'attache aux données de la fonctionnelle $J(u)$ disparaît. En réalité, ce terme est intrinsèque à la méthode puisqu'un contrôle de fidélité est effectué sur les coefficients DCT afin de rester suffisamment proche de l'original, c'est-à-dire dans le même intervalle de quantification.

Considérant l'application dans le cadre du codage JPEG, Alter et al. [12] proposent d'adapter le calcul de la variation totale. Afin de réduire plus efficacement l'effet de bloc, la méthode propose de pondérer la variation totale afin d'augmenter l'importance du terme de régularisation au voisinage des frontières entre blocs de l'image. Ainsi, la fonctionnelle à minimiser s'exprime sous la forme :

$$J_\alpha\left(u\left(\mathbf{x}\right)\right) = \int_\Omega \alpha\left(\mathbf{x}\right) \left|\nabla u\left(\mathbf{x}\right)\right| d\mathbf{x} \tag{5.35}$$

où $\alpha\left(\mathbf{x}\right)$ est une fonction de pondération positive définie sur Ω, et qui est plus importante aux bords des blocs qu'en leurs milieux. L'existence et l'unicité de la solution à la minimisation de $J_\alpha\left(u\right)$, en lieu et place de $J(u)$, ont été adressés dans [47].

Des travaux équivalents de T.F. Chan ont été menés pour une reconstruction optimale dans le cadre d'une transformation en ondelettes orthonormées, selon un critère basé sur la variation totale [44, 45, 43].

5.2.6 Codage d'image

Une application intéressante du modèle d'inpainting présenté dans 5.2.4 est d'utiliser cette technique dans un contexte de codage d'image, présentée notamment par T.F. Chan dans [40]. Contrairement aux standards du codage d'image avec perte, JPEG et JPEG2000, qui se basent sur une transformation linéaire de tous les pixels dans un espace plus adapté, l'idée de la méthode de codage présentée ici repose sur un ensemble réduit de pixels codés. Cet ensemble de pixels, notés C, sert de référence pour interpoler les pixels restants de l'image appartenant à l'ensemble $D = \Omega - C$. L'interpolation repose sur la méthode de régularisation par minimisation de la variation totale vue en 5.2.4.1.

L'ensemble D doit correspondre aux zones de l'image qui se prêtent bien au problème de régularisation de 5.2.4. De façon intuitive, la sélection de C s'opère sur les zones lisses de l'image. A l'inverse, l'ensemble C "connu" correspond aux contours de l'image. Pour déterminer C, une simple méthode de détection de contour de type Canny [33] est employée. Suite à cette opération, l'ensemble C des pixels est dilaté pour constituer une information plus dense et exploitable aux abords des contours de l'image. Cela se justifie par le fait que pour interpoler une zone délimitée par un contour, il est nécessaire d'avoir plus d'information que le contour lui même.

La valeur et la position des pixels de C sont codés, et les pixels de D sont calculés au décodeur par interpolation en utilisant la formule 5.28. En résumé, le mécanisme de régularisation est identique à la méthode d'inpainting présentée dans 5.2.4. La différence se situe au niveau du contrôle des pixels à restaurer dans ce contexte de codage : les pixels qui ne peuvent être restaurés par la méthode de régularisation sont codés, les autres sont interpolés.

Cette méthode de compression, illustrée sur la figure 5.7, possède cependant quelques inconvénients majeurs :

- La méthode d'interpolation utilisée donne un rendu extrêmement lisse des zones régularisées, ce qui enlève le caractère naturel des textures de l'image.
- La méthode nécessite de coder à la fois la position et la valeur des pixels C, ce qui peut considérablement affecter l'efficacité de codage. L'article [40] ne propose pas d'outils de codage spécifiques. Un codage de type JBIG [110] pour transmettre le masque binaire de C semble cependant être un outil adapté et suffisamment performant.

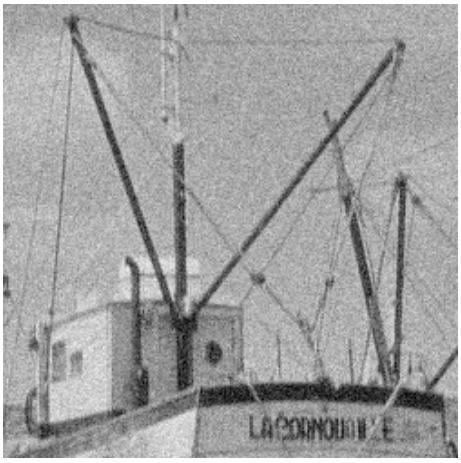
De plus, le type d'image influence l'efficacité de compression de l'algorithme. Cela s'explique facilement par le fait que le nombre de pixels $\mathrm{card}\,(C)$ est lié à la fois à la quantité de contours d'une image et au coût de codage de celle-ci. En d'autres termes, une image plus complexe nécessitera une augmentation du débit pour avoir une qualité de reconstruction suffisante.

Il est intéressant de noter que la méthode de compression proposée ici aboutit à un schéma de codage/décodage asymétrique en terme de complexité, en ce sens où le décodage est d'une complexité nettement supérieure. L'encodage est une succession d'opérations simples : filtrage de Canny, dilatation et codage des pixels. En revanche, le décodage repose sur une interpolation des pixels manquants par une méthode de régularisation variationnelle. Or, nous avons vu que cette méthode est complexe, notamment par le coup d'estimation de la variation totale.

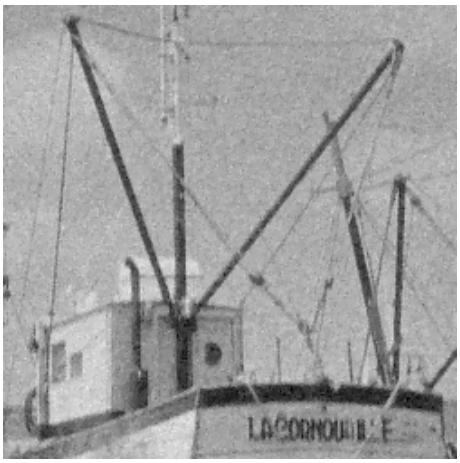
Les articles de Galić et al.[62, 63] présentent une méthode similaire qui code certains pixels épars de l'image et calcule les autres en opérant une diffusion anistropique. Pour coder efficacement les pixels éparpillés dans l'image, la méthode est couplée à un algorithme de codage en *B-tree* triangulaire présenté par Distasi et al. dans[53].

Aujol et al. [17] présentent une méthode de codage d'image différente. En premier lieu, l'image est décomposée selon sa partie structurelle u et sa partie oscillante v (voir 5.2.3). La composante v contenant les éléments oscillants de l'image est codée selon un algorithme classique de compression par ondelettes biorthogonales particulièrement bien adapté à ce contexte d'utilisation [92]. En revanche, la compression de la partie géométrique u est réalisée grâce à un algorithme mieux adapté pour coder les structures d'une image, basé sur les travaux de Cohen et al. [48]. Cette approche, qui adapte le modèle de compression en fonction de la composante traitée, donne de meilleurs résultats que la méthode standard par ondelettes orthogonales.

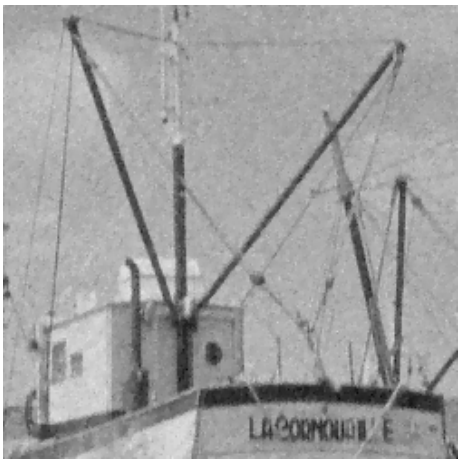
(a) origine



(b) image bruitée ($PSNR = 30,11$)



(c) λ_0 ($PSNR = 32,42$)



(d) λ_1 ($PSNR = 33,57$)

(e) λ_2 ($PSNR = 33,17$)

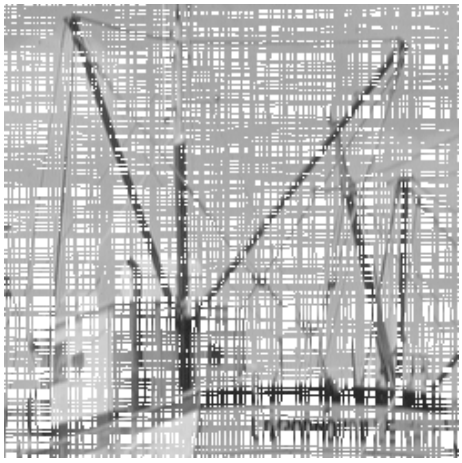
(f) λ_3 ($PSNR = 28,29$)

FIGURE 5.3 – Débruitage d'une image avec la norme TV pour différentes valeurs de λ ($\lambda_i > \lambda_{i+1}$).) avec 100 itérations.

(a) f

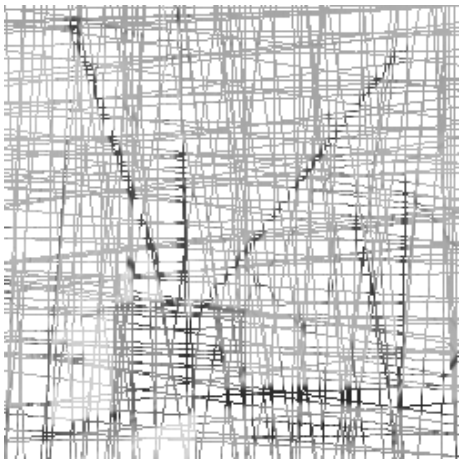
(b) u (c) v

FIGURE 5.4 – Illustration du modèle de décomposition d'image en structure/texture de Vese-Osher [145] avec $f = u + v$ (100 itérations).



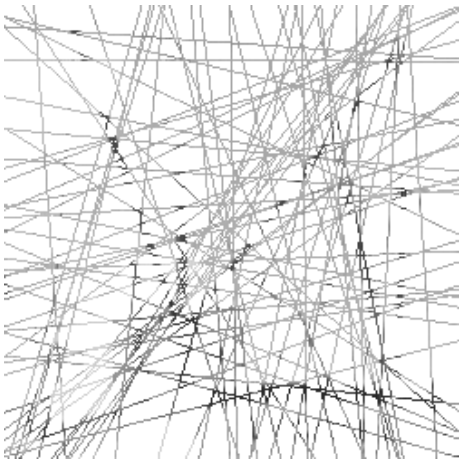
(a) 25% de pixels supprimés

(b) Image restaurée (109 itérations)



(c) 50% de pixels supprimés

(d) Image restaurée (167 itérations)



(e) 75% de pixels supprimés

(f) Image restaurée (250 itérations)

FIGURE 5.5 – Illustration de la méthode d'inpainting en fonction du nombre de pixels disponibles (image d'origine Fig. 5.3).

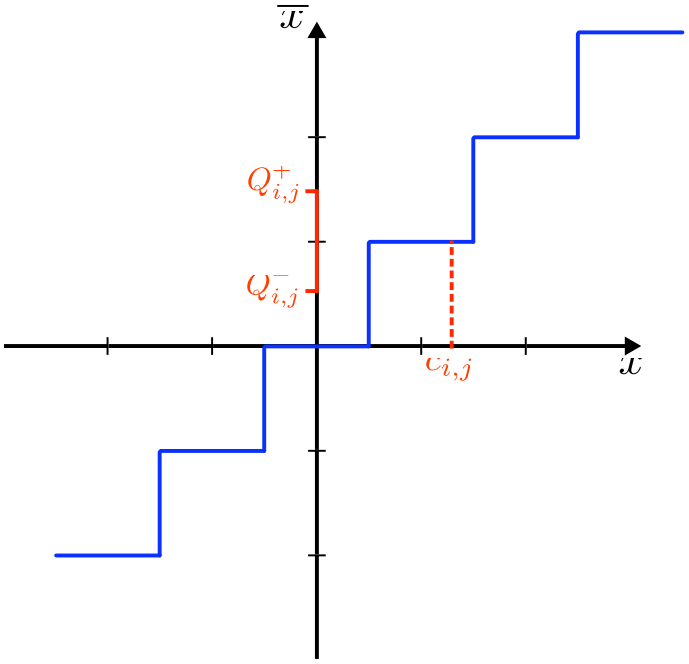
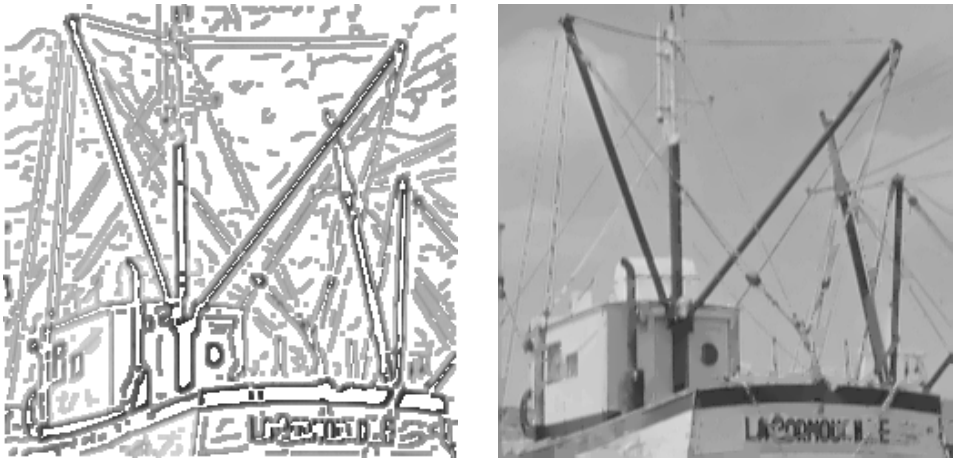
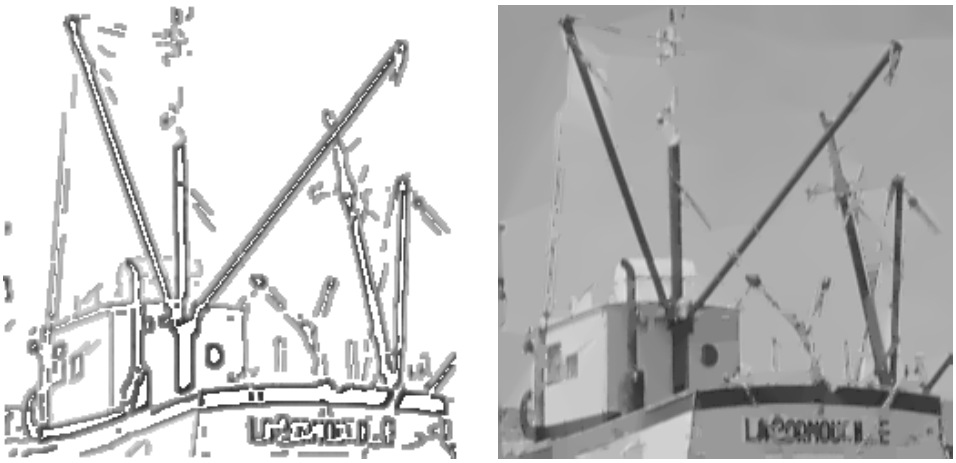


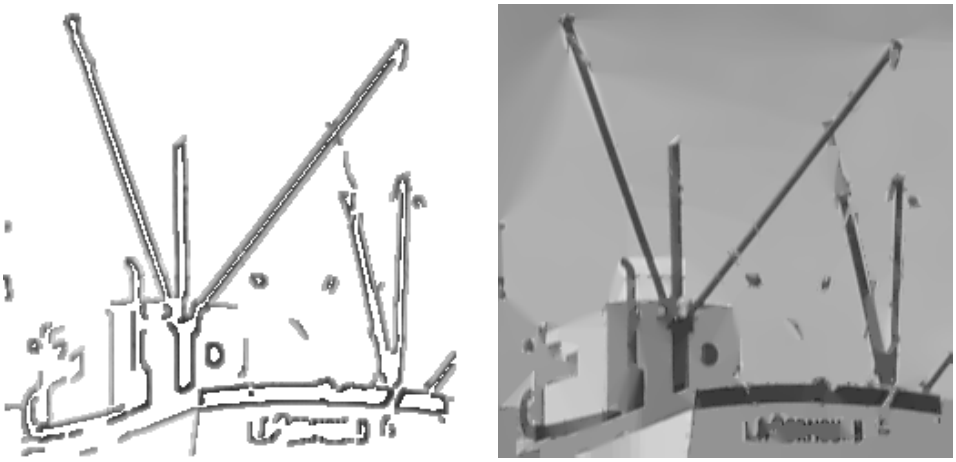
FIGURE 5.6 – Fonction de quantification scalaire uniforme $\overline{x} = Q(x)$. Pour une reconstruction optimale, la contrainte imposée est $\overline{c_{i,j}} \in \left[Q^-_{i,j}, Q^+_{i,j}\right]$.



(a) 50% des pixels codés (b) $PSNR = 37.51$



(c) 25% des pixels codés (d) $PSNR = 35.05$



(e) 15% des pixels codés (f) $PSNR = 33.78$

FIGURE 5.7 – Illustration de la méthode de codage par interpolation des zones lisses. A gauche : les pixels transmis. A droite : l'image décodée. Image source : 5.3a.

5.3 Conclusion

Dans ce chapitre, nous avons étudié un modèle de régularisation dédié à la restauration d'image, formalisé comme un problème inverse mal-posé. Le cadre de l'étude s'est focalisé sur une méthode de résolution variationnelle définie par la minimisation d'une fonctionnelle. Selon le type d'application, cette fonctionnelle est composée de termes différents permettant d'appliquer des contraintes de restauration en fonction des connaissances *a priori* des dégradations subies par l'image. Dans beaucoup de cas, un terme d'attache aux données contraint l'image restaurée à être suffisamment proche de l'image d'origine. Un terme de régularisation permet quant à lui "d'assouplir" l'image de façon à la percevoir de façon plus conforme à la réalité. Comme nous l'avons vu, la norme TV est particulièrement bien adaptée dans ce contexte de restauration d'image puisqu'elle permet de lisser l'image tout en conservant les contours.

La minimisation d'une fonctionnelle conduit à la résolution des équations EDP d'Euler-Lagrange associées. Nous avons vu qu'il est ainsi possible d'utiliser un algorithme de descente de gradient pour converger numériquement vers une solution. Cependant, ces équations sont fortement non-linéaires et difficiles à résoudre. Plusieurs travaux ont été réalisés pour accélérer la résolution de ce type de problème, notamment dans les articles [149, 37, 72, 66].

Autour du modèle de régularisation étudié, la littérature est composée de plusieurs autres méthodes de résolution variationnelles. Les travaux de Tschumperlé et al. [142] (puis [143, 144]) généralisent plusieurs approches concernant la régularisation d'image basée sur la diffusion d'équation anisotropique. Ils proposent ainsi un cadre qui unifie plusieurs méthodes de régularisation existantes, en particulier l'approche par minimisation de fonctionnelles, évoquée dans ce chapitre, mais aussi l'expression de la divergence et la formulation par laplacien orienté. Les applications proposées sont nombreuses, dont l'amélioration d'une image compressée avec perte selon le standard JPEG. Les dégradations observables par ce type d'encodage à des débits faibles sont fortement atténuées grâce à la méthode proposée par Tschumperlé, voire indétectables.

Différentes utilisations du modèle de régularisation d'image ont été présentées et illustrées dans ce chapitre. Plusieurs d'entre elles nous ont inspirés pour mener les travaux présentés dans le chapitre suivant. En particulier, les méthodes de restauration des coefficients en ondelettes détériorés nous ont montré qu'il était possible d'assimiler la régularisation d'image dans le domaine spatial avec un contexte de restauration des coefficients dans un espace transformé. De ce fait, il nous est apparu envisageable d'agrémenter un schéma de codage classique basé sur une transformation linéaire par une méthode de restauration des coefficients dans le but d'en améliorer les performances en compression ; c'est l'objet du chapitre suivant.

CHAPITRE 6

Nouvelle méthode de codage vidéo par prédiction dans le domaine transformé

Sommaire

Ce chapitre est consacré aux travaux de thèse réalisés dans le but de proposer un nouvel outil de codage vidéo afin d'améliorer les performances en compression. La section suivante positionne ce travail de recherche par rapport à l'état de l'art présenté dans le chapitre précédent. L'idée sous-jacente est d'utiliser les méthodes de régularisation d'image afin de prédire une quantité de données au lieu de la coder, ceci afin d'améliorer les performances en compression des algorithmes de codage d'image et de vidéo. Puis nous étudierons le modèle de restauration proposé dans un cadre théorique. La première application directe va consister à utiliser ce modèle pour restaurer des coefficients DCT dégradés, par exemple à cause d'une perte de données pendant la transmission ou d'un procédé de compression destructive. Ensuite, nous exposerons comment le modèle de restauration peut être intégré dans un encodeur d'image de type JPEG et nous présenterons les gains obtenus en terme de

débit. Enfin, l'objectif final de ce chapitre sera d'adapter ce mécanisme aux spécificités du schéma de codage vidéo de type H.264/AVC.

6.1 Introduction

L'encodeur H.264/AVC est un standard vidéo très performant qui permet d'obtenir un ratio de compression conséquent. Comme nous l'avons vu dans le premier chapitre, le standard repose sur un partitionnement par blocs de l'image, à l'instar de JPEG. Pour rappel, chaque bloc est prédit par interpolation des pixels voisins disponibles (prédiction intra-image) ou à partir d'images précédemment encodées (prédiction inter-image). Le résidu, c'est-à-dire l'erreur de prédiction, est ensuite transformé, quantifié et codé pour obtenir un flux binaire. L'étape de transformation des blocs résiduels est basée sur une DCT, bien que ce soit une approximation pour en simplifier la complexité. C'est pourquoi dans ce chapitre nous nous focaliserons uniquement sur le cas de la transformée DCT, bien que des parallèles soient possibles avec d'autres types de transformation (notamment les ondelettes).

6.1.1 Positionnement par rapport à l'état de l'art

Nous avons vu dans le chapitre précédent que des travaux avaient été menés dans l'optique de proposer de nouveaux schémas de codage ou d'améliorer ceux existants, ceci afin d'accroître le ratio de compression. La première application dans ce contexte est la reconstruction optimale d'une image compressée avec pertes, dans le but d'augmenter la qualité de l'image décodée (voir 5.2.5), en prenant en compte les dégradations réalisées sur l'image par le procédé d'encodage. La reconstruction optimale s'opère dans un contexte de post-traitement effectué après le décodage et n'est pas partie intégrante de l'algorithme de codage/décodage. De ce point de vue, il ne contribue pas à diminuer le volume nécessaire de données transmises mais est utile pour "masquer" les artefacts visuels gênants qui apparaissent sur les images fortement compressées.

Le deuxième type d'application présentée consiste à ne pas coder les pixels d'une image qui peuvent être restaurés par une méthode de régularisation (voir 5.2.6). Ainsi, les pixels codés servent de support à la restauration de l'information manquante. Dans ce cas, nous avons vu qu'il est intéressant de ne coder que les portions d'images contenant les contours et leurs voisinages. Les inconvénients de ce type d'approche sont multiples. Tout d'abord, il est nécessaire de connaître, pour chaque pixel codé, sa position dans l'image, ce qui au final s'avère très coûteux en terme de débit. De plus, cette méthode est difficilement conciliable avec les procédés classiques de transformation et quantification très répandus dans un contexte de codage d'image.

Au contraire, nous étudions une nouvelle approche qui consiste, dans un premier temps, à intégrer un procédé de régularisation d'image directement au sein d'un codeur image de type JPEG. Ensuite, nous nous intéresserons au cas particulier du codage vidéo de type H.264/AVC. De ce fait, nous bénéficions des mécanismes introduits dans ces normes et qui ont démontré leur efficacité en terme de compression. Nos travaux se concentrent sur la prédiction de coefficients DCT associés aux pixels d'une image. Cette étape de prédiction s'inspire des méthodes de restauration d'image vues précédemment : nous cherchons à estimer certains coefficients DCT de façon à ce qu'ils "corrigent" l'image selon des critères basés sur la variation totale. Le fait de travailler dans le domaine transformé plutôt que dans le domaine des pixels change la nature du problème de restauration, puisque qu'un seul

coefficient DCT dégradé affecte plusieurs pixels. Ainsi, les techniques d'interpolation géométrique pour restaurer les pixels manquants d'une image ne peuvent s'appliquer directement dans ce cas. D'un autre côté, l'interpolation des coefficients directement dans le domaine DCT est également problématique puisque les coefficients sont justement calculés pour décorréler le signal. Pour ces raisons, nos travaux fixent un modèle basé sur des changement successifs entre le domaine spatial, lié aux problématiques de régularisation d'image, et le domaine DCT, où s'opère la prédiction des coefficients.

6.1.2 Introduction au formalisme

En premier lieu, nous nous intéressons au formalisme de la DCT-2D appliquée sur des blocs carrés de petites tailles, telle qu'elle est utilisée dans un contexte de codage d'image [118]. En effet, la DCT est réversible et possède une propriété de regroupement de l'énergie qui s'adapte très bien aux images naturelles, permettant ainsi de réduire le volume de données nécessaires tout en conservant une qualité de reconstruction acceptable pour le système visuel humain.

La DCT s'applique sur une matrice carré $\mathbf{u}_B$ de pixels de taille $N_\mathcal{B} \times N_\mathcal{B}$, soit un sous-ensemble d'une image en niveaux de gris u (définie sur Ω) repéré par les coordonnées B, et engendre α_B, une matrice de coefficients DCT de dimension identique à $\mathbf{u}_B$. Soit $\mathbf{A}$ la matrice de la transformation DCT telle que :

$$\alpha_B = \mathbf{A}\mathbf{u}_B\left(\mathbf{A}^T\right) \tag{6.1}$$

$$\mathbf{u}_B = \left(\mathbf{A}^T\right)\alpha_B\mathbf{A} \tag{6.2}$$

où $\mathbf{A}^T$ représente la matrice transposée de $\mathbf{A}$. La matrice $\mathbf{A}$, de taille $N_\mathcal{B} \times N_\mathcal{B}$, est définie par :

$$A_{ij} = C_i \cos\frac{(2j+1)\,i\pi}{2N_\mathcal{B}} \tag{6.3}$$

où A_{ij} représente l'élément à la position (i,j) dans la matrice $\mathbf{A}$ avec $0 \leq i,j < N_\mathcal{B}$, et C_i est un paramètre défini pour que :

$$C_i = \begin{cases} \sqrt{\frac{1}{N_\mathcal{B}}} & si \quad i = 0 \\ \sqrt{\frac{2}{N_\mathcal{B}}} & sinon \end{cases} \tag{6.4}$$

Les équations peuvent 6.1 et 6.2 peuvent se réécrire :

$$\alpha_{B\vec{k}} = C_i C_j \sum_{x=0}^{N_{\mathcal{B}-1}} \sum_{y=0}^{N_{\mathcal{B}-1}} u_{B\vec{z}} \cos\left(\frac{(2x+1)\,i\pi}{2N_\mathcal{B}}\right)\cos\left(\frac{(2y+1)\,j\pi}{2N_\mathcal{B}}\right) \tag{6.5}$$

$$u_{B\vec{z}} = \sum_{i=0}^{N_{\mathcal{B}-1}} \sum_{j=0}^{N_{\mathcal{B}-1}} C_i C_j \alpha_{B\vec{k}} \cos\left(\frac{(2x+1)\,i\pi}{2N_\mathcal{B}}\right)\cos\left(\frac{(2y+1)\,j\pi}{2N_\mathcal{B}}\right) \tag{6.6}$$

où $\vec{k} = \begin{pmatrix} i \\ j \end{pmatrix}$ correspond à la position du coefficient DCT dans le bloc α_B et $\vec{z} = \begin{pmatrix} x \\ y \end{pmatrix}$ la position du pixel dans le bloc $\mathbf{u}_B$.

6.2 Modèle de restauration des coefficients DCT

Le modèle de restauration mis au point ici est basé sur une méthode de régularisation variationnelle contrainte. La particularité de la méthode proposée réside dans le fait que la restauration s'applique sur des coefficients DCT dégradés, alors que la contrainte de régularisation, basée sur la minimisation de la variation totale, s'exprime dans le domaine spatial.

Soit $\beta_{B\vec{k}}$ la version dégradée des coefficients DCT $\alpha_{B\vec{k}}$. Dans cette section, nous considérons que les coefficients dégradés correspondent à des coefficients manquants, soit $\beta_{B\vec{k}} = 0 \, \forall \vec{k} \in I_{DCT}$, avec I_{DCT} l'ensemble des coefficients manquants du bloc β_B. D'autres types de dégradations peuvent également être envisagés. Par exemple, nous verrons par la suite une application qui restaure des coefficients DCT dégradés suite à une étape quantification, ce qui fait le rapprochement avec les travaux de [12] et les méthodes de reconstruction optimale.

La variation totale de l'image dans le domaine direct à partir des coefficients $\beta_{B\vec{k}}$ est exprimée par :

$$\left\| u_{B\vec{z}}\left(\beta_{B\vec{k}}\right) \right\|_{TV} = \int_{\Omega} \left| \nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right) \right| d\vec{z} \tag{6.7}$$

L'intégrale porte sur l'ensemble Ω (le support de l'image dans $\mathbb{R}^2$) et où u est considérée nulle en dehors du bloc B. Nous appliquons la contrainte suivante :

$$\beta_{B\vec{k}} = \alpha_{B\vec{k}} \text{ pour tout } \vec{k} \notin I_{DCT} \tag{6.8}$$

Cette contrainte nous assure que seuls les coefficients perdus seront restaurés. L'Eq. 6.15 correspond au terme de régularisation que nous cherchons à minimiser, et que nous déclinons tel que :

$$\frac{\partial \left\| u_{B\vec{z}}\left(\beta_{B\vec{k}}\right) \right\|_{TV}}{\partial \beta_{B\vec{k}}} = \int_{\Omega} \frac{\partial \left| \nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right) \right|}{\partial \beta_{B\vec{k}}} d\vec{z} \tag{6.9}$$

En utilisant la dérivée d'une composition de fonctions dérivables, nous obtenons :

$$\begin{array}{rcl} \frac{\partial \left\| u_{B\vec{z}}\left(\beta_{B\vec{k}}\right) \right\|_{TV}}{\partial \beta_{B\vec{k}}} & = & \int_{\Omega} \frac{\nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)}{\left| \nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right) \right|} \cdot \frac{\partial \nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)}{\partial \beta_{B\vec{k}}} d\vec{z} \\ & = & \int_{\Omega} \frac{\nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)}{\left| \nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right) \right|} \cdot \nabla_{\vec{z}} \frac{\partial u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)}{\partial \beta_{B\vec{k}}} d\vec{z} \end{array} \tag{6.10}$$

D'après la définition de la transformée DCT inverse 6.6, nous avons :

$$\frac{\partial u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)}{\partial \beta_{B\vec{k}}} = \phi_{B\vec{z},\vec{k}} \tag{6.11}$$

où $\phi_{B\vec{z},\vec{k}}$ caractérise le noyau de la transformée DCT appliqué au bloc B, soit :

$$\frac{\partial \left\| u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)\right\|_{TV}}{\partial \beta_{B\vec{k}}} = \int_{\Omega} \frac{\nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)}{\left|\nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)\right|} \cdot \nabla_{\vec{k}} \phi_{B\vec{z},\vec{k}} d\vec{z} \tag{6.12}$$

En appliquant une intégration par partie, nous obtenons :

$$\begin{aligned} \int_{\Omega} \frac{\nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)}{\left|\nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)\right|} \cdot \nabla_{\vec{k}} \phi_{B\vec{z},\vec{k}} d\vec{z} = & \left[\nabla_{\vec{k}} \cdot \left(\frac{\nabla_{\vec{k}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)}{\left|\nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)\right|}\right) \phi_{B\vec{z},\vec{k}}\right]_{0}^{\Omega} \\ & - \int_{\Omega} \left(\nabla_{\vec{k}} \cdot \frac{\nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)}{\left|\nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)\right|}\right) \phi_{B\vec{z},\vec{k}} d\vec{z} \end{aligned} \tag{6.13}$$

Nous considérons que la transformée DCT est à support borné, c'est-à-dire que le noyau $\phi_{B\vec{z},\vec{k}}$ est nul en dehors du bloc B. Ce qui s'exprime par :

$$\left[\nabla_{\vec{k}} \cdot \left(\frac{\nabla_{\vec{k}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)}{\left|\nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)\right|}\right) \phi_{B\vec{z},\vec{k}}\right]_{0}^{\Omega} = 0 \tag{6.14}$$

Au final, la dérivée partielle de la variation totale du bloc $\mathbf{u}_B$ s'exprime par :

$$\frac{\partial \left\| u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)\right\|_{TV}}{\partial \beta_{B\vec{k}}} = -\int_{\Omega} \left[\nabla_{\vec{k}} \cdot \frac{\nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)}{\left|\nabla_{\vec{z}} u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)\right|}\right] \phi_{B\vec{z},\vec{k}} d\vec{z} \tag{6.15}$$

Le terme $\nabla \cdot [\nabla u / |\nabla u|]$ (la divergence du gradient normalisé) exprime la courbure des lignes de niveau de l'image (voir 5.1.3.2).

Le modèle de régularisation proposé consiste à minimiser la variation totale de l'image, soit :

$$\min_{\beta_{B\vec{k}}, \vec{k} \in I_{DCT}} \left\{ J\left(u_{B\vec{z}}, \beta_{B\vec{k}}\right) = \left\| u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)\right\|_{TV} \right\} \tag{6.16}$$

La fonctionnelle J possède un minimum si l'équation d'Euler-Lagrange associée est vérifiée, c'est-à-dire :

$$\frac{\partial \left\| u_{B\vec{z}}\left(\beta_{B\vec{k}}\right)\right\|_{TV}}{\partial \beta_{B\vec{k}}} = 0 \tag{6.17}$$

L'Eq. 6.15 nous montre que converger vers une solution au problème de restauration revient à minimiser la courbure, donc les variations du gradient, transformée dans le domaine DCT. Avec cette nouvelle formulation, la contrainte $\beta_{B\vec{k}} = \alpha_{B\vec{k}}$ pour tout $\vec{k} \in I_{DCT}$ peut être respectée. Au final, nous obtenons un modèle qui permet de régulariser l'image tout en conservant intact les coefficients du domaine DCT non dégradés.

Algorithme 6.1 Méthode générique de reconstruction optimale d'images compressées avec perte par minimisation contrainte de la variation totale.

1. Soient α les coefficients dégradés de l'image u
2. $\alpha_{B\vec{k}} = 0 \; \forall B\vec{k} \in I_{DCT}$, I_{DCT} la position des coefficients perdus
3. Soit $n = 0$ l'itération courante et γ_n le pas de descente de gradient fonction de n
4. Soit $\beta^{new} = \alpha$ et $\beta^{old} = 0$
5. Soit K le nombre maximum d'itérations et ϵ la différence minimale entre deux images successives
6. **Tant que** $n < K$ **et** $\left\|\beta^{new} - \beta^{old}\right\|_2 > \epsilon$**, faire**
7. $\beta^{old} = \beta^{new}$
8. Pour tous les coefficients $\vec{k}$ de tous les blocs B appartenant à I_{DCT}, faire

$$\beta^{new}_{B\vec{k}} = \beta^{old}_{B\vec{k}} + \gamma_n \left[DCT\left(curv\left(IDCT\left(\beta^{old}_{B\vec{k}} \right) \right) \right) \right] \tag{6.18}$$

9. Calculer $n = n + 1$
10. **Fin de la boucle, l'image décodée est donnée par** $u = IDCT(\beta^{new})$

6.3 Amélioration d'image par restauration des coefficients DCT

Nous mettons à contribution le modèle exposé précédemment pour restaurer des images dont les coefficients ont été endommagés. Pour cela, nous considérons deux types d'applications :

– La première simule une perte aléatoire de coefficients DCT calculés à partir d'une image. L'objectif est de régulariser l'image en ne restaurant que les coefficients DCT perdus et dont la position est supposée connue (l'ensemble I_{DCT}). Il s'agit de l'implémentation directe du modèle présenté en 6.2.

– La deuxième s'utilise dans un contexte de codage d'image avec perte. Il s'agit de régulariser l'image en modifiant les coefficients DCT issus de la quantification inverse. La contrainte du modèle de restauration est qu'un coefficient DCT doit être contenu dans son intervalle de quantification. Ce cas d'utilisation rejoint les travaux de [12].

6.3.1 Restauration de coefficients DCT aléatoirement perdus

Nous testons ici la capacité du modèle à restaurer des coefficients DCT perdus selon un critère de qualité objectif. Pour cet exercice, nous n'employons pas de procédé de quantification. L'algorithme de restauration, basé sur une descente de gradient, est présenté en 6.1. Nous supposons que l'ensemble I_{DCT} des coefficients perdus est connu. La fonction *curv* calcule la courbure de l'image (voir 5.1.3.2). Les fonctions DCT et $IDCT$ calculent respectivement la transformée DCT et la transformée DCT inverse pour un bloc ou un ensemble de blocs (voir 6.1.2).

Nous utilisons un partitionnement par blocs de 8×8 pixels avec une transformée DCT entière donnée par 6.3. Le paramètre γ_n, le pas de descente de gradient, est constant et fixé à 1. Le nombre d'itérations pour converger vers un état stable pourrait être optimisé en paramétrant γ_n de façon

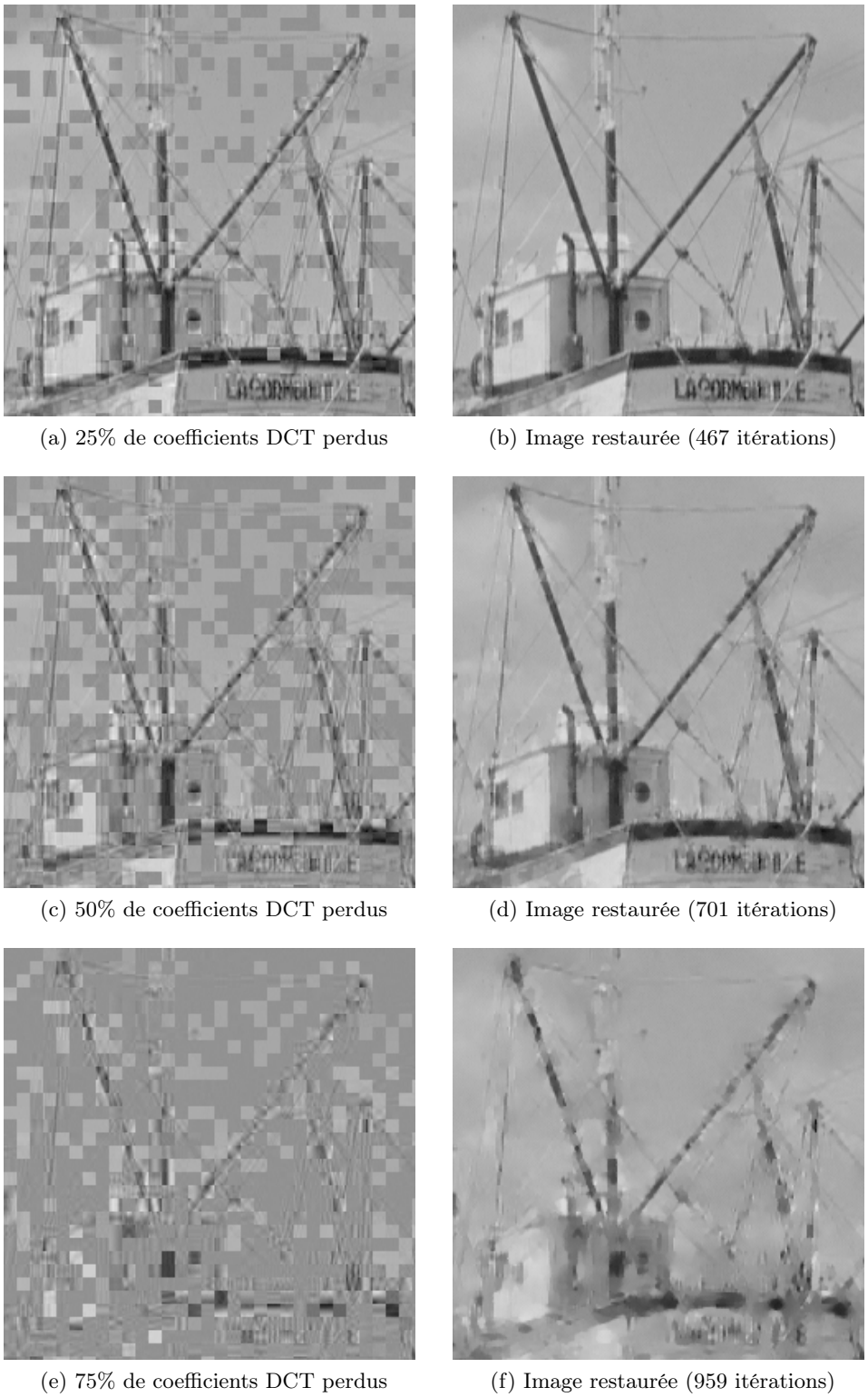
moins arbitraire. Cependant, ce n'est pas l'objectif de cet exercice où nous recherchons avant tout une solution optimale pour mesurer l'efficacité maximale de la méthode. La figure 6.1 illustre les résultats obtenus. Les gains de qualité objective (PSNR) sont représentés sur la figure 6.2. Il est intéressant de noter que subjectivement et objectivement, les images obtenues après la perte de coefficients DCT apparaissent fortement dégradées, même lorsque seulement 25% des coefficients sont perdus. Les coefficients DC perdus sont initialisés pour que la moyenne des pixels du bloc soit égale à 127, ce qui entraîne dans ce cas un effet de bloc. Les coefficients AC perdus produisent quant à eux des altérations de l'image aux niveaux des contours. Pour 25% des coefficients DCT perdus, la méthode de restauration donne les meilleurs résultats (+12.36 dB en PSNR). L'écart se réduit quand le nombre de coefficients perdus augmente, jusqu'à $+6,79$ dB pour 75% de coefficients supprimés. En effet, le procédé de restauration se base sur l'information connue pour estimer le reste, donc la qualité de la restauration dépend directement de l'information initiale disponible (tous les coefficients n'appartenant pas à I_{DCT}).

6.3.2 Amélioration des images décodées

L'exemple d'utilisation présenté dans cette section a pour objectif de montrer que le modèle théorique introduit en 6.2 peut être décliné selon le type d'application souhaité. Cette nouvelle application se place dans le cadre des méthodes de reconstruction optimale présentées en 5.2.5, et notamment de l'algorithme introduit en 5.1. Nous souhaitons régulariser l'image décodée pour supprimer les artefacts liés à l'algorithme de compression et imputés à l'étape de quantification. Pour cela, nous recourons au modèle de minimisation de la variation totale préalablement présenté (6.16). La méthode d'amélioration de l'image est également contrainte par un terme d'attache aux données qui contient les coefficients DCT à l'intérieur de leurs intervalles de quantification. Grâce à ce mécanisme, un coefficient DCT décodé peut osciller à l'intérieur de ses bornes de quantification $[Q^{-}_{B\vec{k}}, Q^{+}_{B\vec{k}}]$ afin de générer une image la plus lisse possible (au sens de la variation totale).

Les résultats de l'algorithme 5.1 sont présentés sur les figures 6.3 et 6.4. Les images améliorées présentent une meilleure qualité objective que l'image décodée de façon relativement constante, avec un gain moyen de $+0.7dB$ en PSNR. Le nombre d'itérations nécessaires est particulièrement faible en comparaison de la méthode de restauration de coefficients DCT perdus présentée en 6.3.1. En effet, dans ce cas d'utilisation, la solution optimale est proche de l'originale à cause de la contrainte de proximité des coefficients DCT améliorés (restreints à l'intervalle de quantification des coefficients d'origine). Visuellement, les images améliorées montrent une atténuation des effets de blocs qui se manifestent d'autant plus à bas-débit.

En conclusion, la méthode présente le double avantage de pouvoir améliorer la qualité d'une image décodée sans recourir à une modification du standard de compression d'image. En effet, le post-traitement est effectué uniquement au niveau du décodeur, donc il n'est pas nécessaire de modifier l'encodeur.



(a) 25% de coefficients DCT perdus

(b) Image restaurée (467 itérations)

(c) 50% de coefficients DCT perdus

(d) Image restaurée (701 itérations)

(e) 75% de coefficients DCT perdus

(f) Image restaurée (959 itérations)

FIGURE 6.1 – Illustration de l'algorithme de restauration des coefficients DCT (6.1).

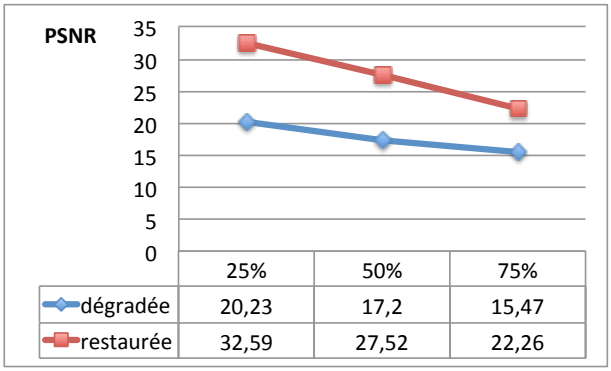
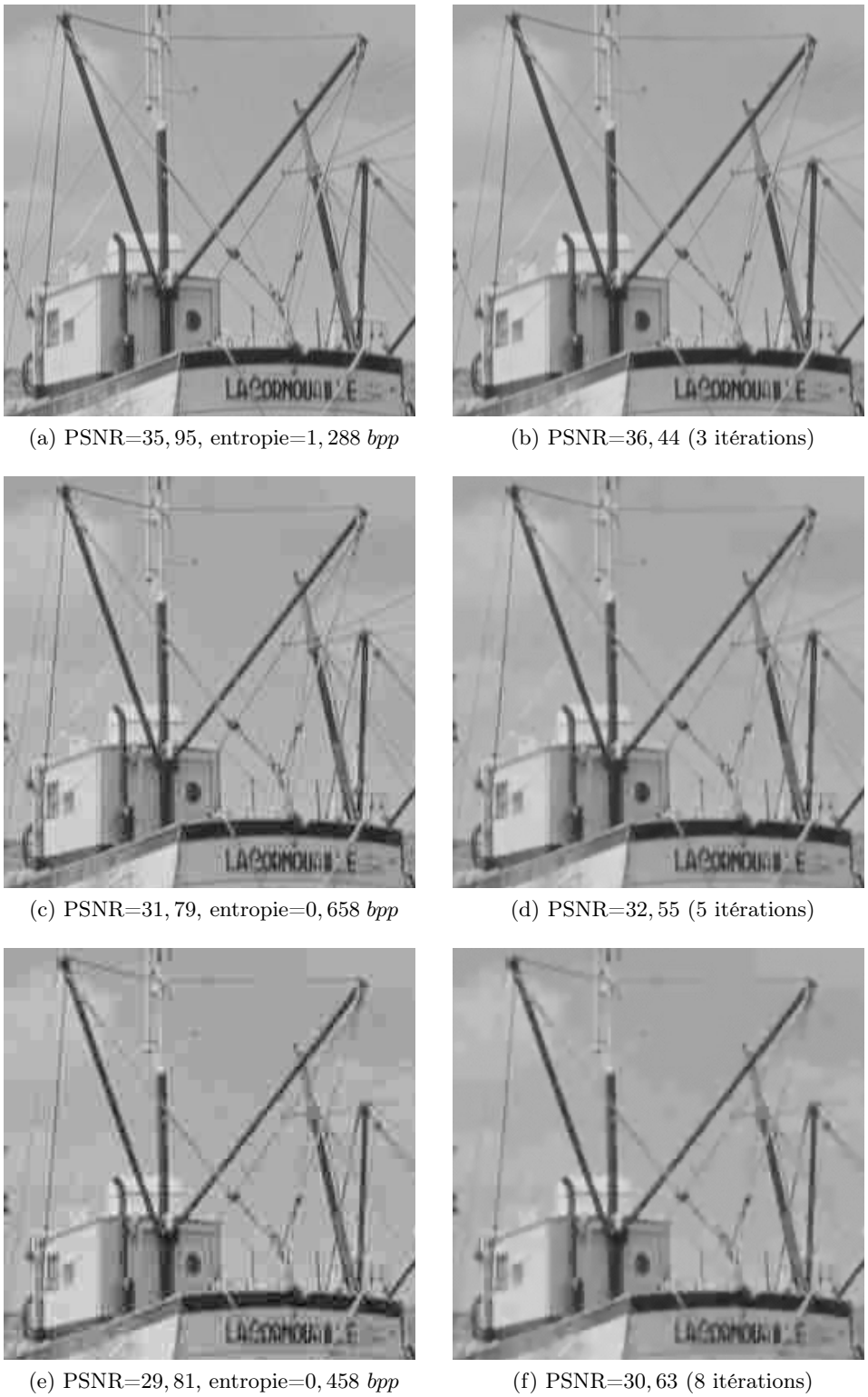


FIGURE 6.2 – PSNR des images dégradées/restaurées de 6.1 en fonction du pourcentage de coefficients DCT perdus.



(a) PSNR=35, 95, entropie=1, 288 *bpp* (b) PSNR=36, 44 (3 itérations)

(c) PSNR=31, 79, entropie=0, 658 *bpp* (d) PSNR=32, 55 (5 itérations)

(e) PSNR=29, 81, entropie=0, 458 *bpp* (f) PSNR=30, 63 (8 itérations)

FIGURE 6.3 – Illustration de l'algorithme de reconstruction optimale des coefficients DCT. Colonne de gauche : l'image décodée. Colonne de droite : image décodée puis améliorée selon la méthode de reconstruction optimale.

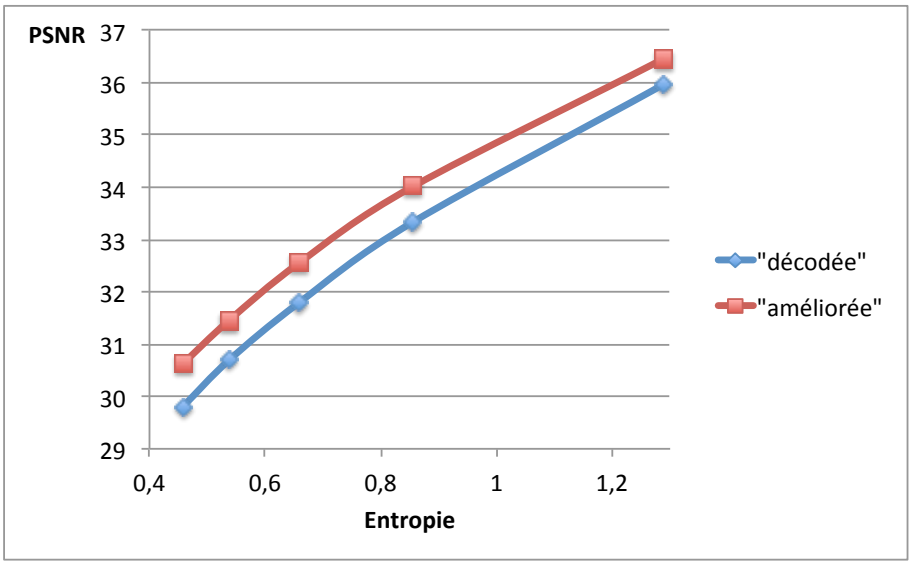


FIGURE 6.4 – PSNR en fonction de l'entropie (en bpp) des images décodées et améliorées selon la méthode de reconstruction optimale.

6.4 Codage d'image fixe par prédiction des coefficients DCT

Nous allons étudier comment adapter la méthode de restauration des coefficients DCT, vue en 6.3.1, afin d'améliorer les performances en compression d'un codeur d'image fixe de type JPEG. Nous faisons l'hypothèse qu'une partie des coefficients DCT peut être restaurée au décodeur sans dégrader l'image et donc sans nécessiter leur codage, assurant ainsi une compression plus efficace. Nous présentons un modèle de prédiction DCT intra-bloc, basé sur l'idée qu'un ensemble "connu" (appelé I_O) des coefficients DCT d'un bloc est utilisé pour prédire l'ensemble complémentaire des coefficients DCT inconnus du même bloc (appelé I_{DCT}).

6.4.1 Schéma de codage

La méthode de restauration des coefficients DCT (section 6.3.1) est utilisée en tant que nouvelle méthode de prédiction et est introduite dans le chaînage des étapes du schéma de codage JPEG. Chaque bloc u_B de l'image subit une DCT afin d'obtenir le bloc DCT α_B correspondant. Les coefficients DCT $\alpha_{B\vec{k}}$ sont répartis en deux ensembles complémentaires I_O et I_{DCT}, les coefficients du premier ensemble servant de support pour la prédiction des coefficients du deuxième. Les coefficients de l'ensemble I_{DCT} ne sont donc pas directement codés/décodés, ils sont dans un premier temps prédits. Au codeur, l'erreur de prédiction est transmise comme indiqué sur la figure 6.5. Au décodeur, le même procédé de prédiction est utilisé, puis le décodage de l'erreur de prédiction permet de reconstruire les coefficients de l'ensemble I_{DCT} (figure 6.6). Pour prédire efficacement les coefficients DCT de I_{DCT} via la méthode de restauration proposée dans 6.3.1, nous recourons aux blocs voisins précédemment codés pour calculer la courbure locale de l'image, permettant ainsi de lisser le bloc courant à partir des pixels adjacents. Ce schéma de codage présente l'intérêt d'aboutir à une image décodée strictement identique par rapport au codeur JPEG standard. A PSNR équivalent, nous cherchons à réduire le débit des images compressées.

La difficulté principale de cette approche est de définir l'ensemble I_{DCT} de façon optimale. Idéalement, I_{DCT} doit correspondre aux coefficients DCT :

– qui peuvent être correctement prédits en utilisant l'algorithme de restauration vu en 6.3.1 ;
– qui possèdent une énergie non négligeable.

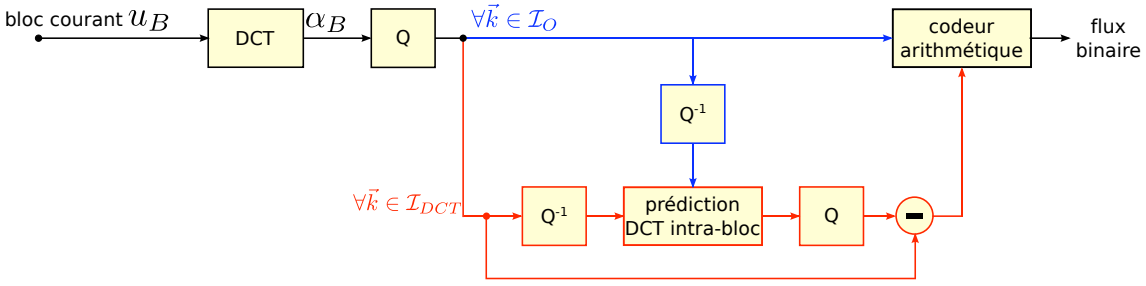


FIGURE 6.5 – Schéma d'un encodeur d'image fixe basé sur JPEG et intégrant une méthode de prédiction intra-bloc des coefficients DCT.

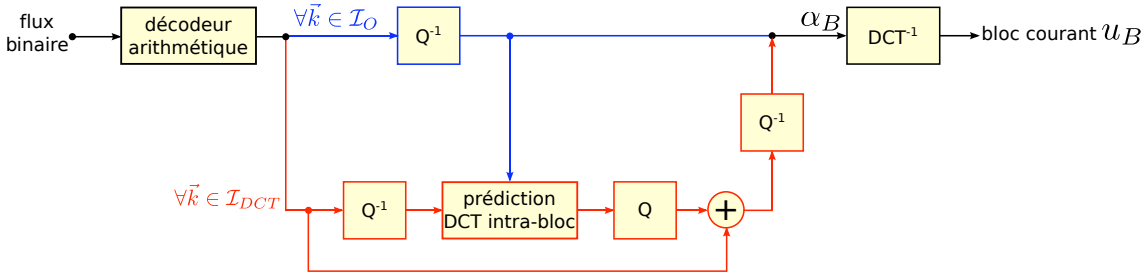


FIGURE 6.6 – Schéma d'un décodeur d'image fixe basé sur JPEG et intégrant une méthode de prédiction intra-bloc des coefficients DCT.

Le premier point est propre à la notion de compression. Pour réduire le débit d'information, l'erreur de prédiction codée doit être globalement plus faible que l'information d'origine non prédite. Il est dans ce cas nécessaire d'utiliser une méthode de prédiction statistiquement efficace.

Le deuxième point est primordial si nous voulons réduire significativement la quantité d'information à transmettre. Dans un contexte de codage, il n'est pas intéressant de prédire des coefficients DCT d'origine nuls ; il est en effet plus avantageux de prédire efficacement les coefficients DCT de forte amplitude afin d'améliorer le facteur de compression. Nous observons ici une contradiction entre ces deux critères puisque la suppression d'un coefficient de forte amplitude va dégrader l'image, diminuant globalement la qualité de la prédiction : il s'agit d'un principe de compromis débit/distorsion. Nous verrons dans les expérimentations suivantes la configuration qui donne les meilleurs résultats.

Deux solutions s'offrent à nous pour définir l'ensemble I_{DCT} (et par complémentarité l'ensemble I_O) :

- Soit l'ensemble I_{DCT} est statique et prédéfini au codeur et au décodeur.
- Soit l'ensemble I_{DCT} est calculé (au niveau image ou bloc) afin d'obtenir le compromis débit/distorsion optimal, et donc améliorer le facteur de compression de l'image. Dans ce cas, il est en revanche indispensable de transmettre la configuration des coefficients DCT prédits, pour que le décodeur puisse réaliser l'opération de prédiction sur les mêmes coefficients.

Au vu des premiers résultats expérimentaux, nous avons choisi de suivre la première piste. En effet, la configuration optimale de I_{DCT} et I_O ne varie pas (ou très peu) d'un bloc à l'autre et d'une image à l'autre. L'utilisation d'ensembles statiques permet ainsi d'économiser une information supplémentaire à transmettre.

Enfin, notons sur 6.5 et 6.6 que si l'ensemble des coefficients prédits I_{DCT} est vide, alors nous exploitons un schéma de codage JPEG classique décomposé en trois étapes conventionnelles : transformation, quantification et codage arithmétique.

6.4.2 Expérimentations et résultats par rapport à la norme JPEG

Nous nous sommes basés sur la librairie *libjpeg* (http ://www.ijg.org/), soutenue par le groupe indépendant JPEG, pour effectuer ces expérimentations. Les tests ont été réalisés sur un ensemble de huit images monochromes avec, pour chaque image, trois qualités différentes données par un paramètre d'encodage compris entre 0 et 100, où 100 représente la meilleure qualité (et le débit le plus élevé). Pour chaque bloc 8×8, la position d'un coefficient DCT est donnée par ses coordonnées $c_{ij} =$

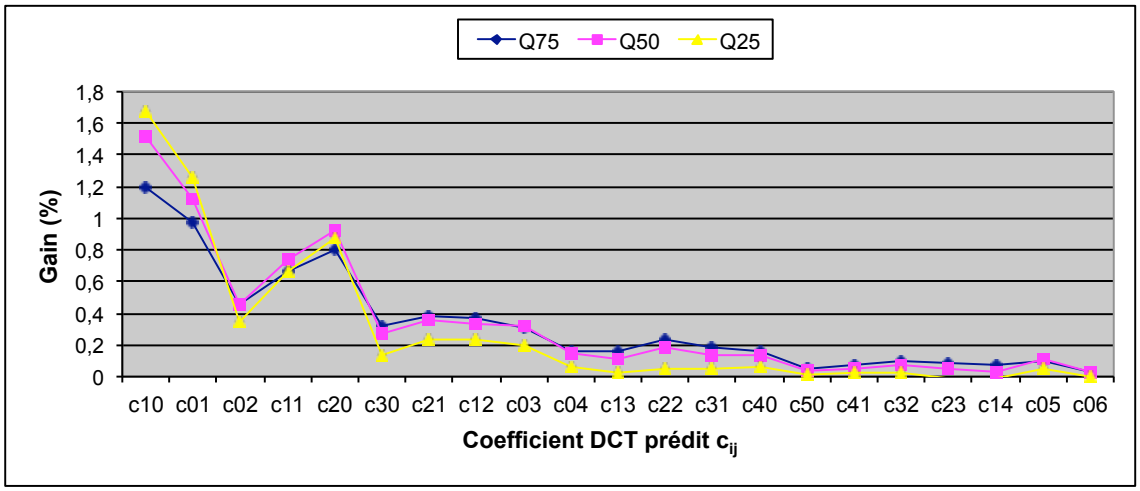


FIGURE 6.7 – Gain moyen en débit (réf : JPEG) en fonction d'un coefficient c_{ij} prédit et d'une qualité de codage $Q \in [25, 50, 75]$.

(i, j), où $0 \leq i, j < 8$. Le codeur arithmétique utilise un code de Huffman optimisé qui nécessite deux passes d'encodage, ainsi que la transmission au décodeur des tables de Huffman correspondantes. Ce choix d'implémentation apporte une solution au fait que, statistiquement, l'erreur de prédiction est différente du coefficient DCT d'origine, et donc est inadaptée aux tables de Huffman prédéfinies dans JPEG.

A travers ces expérimentations, nous cherchons à définir l'ensemble I_{DCT} optimal, c'est-à-dire celui qui donne les meilleurs taux de compression par rapport à JPEG. Dans un premier temps, un seul coefficient à la position c_{ij} est prédit (où $c_{ij} \in I_{DCT}$) pour tous les blocs de l'image, à l'exception du coefficient DC (qui a un mécanisme propre de prédiction dans le standard JPEG). La moyenne des résultats pour l'ensemble des huit images est illustrée sur la figure 6.7. Seuls les 21 premiers coefficients, dans la représentation *zigzag* (2.4) des blocs, sont représentés sur la figure (coefficients correspondant aux plus basses fréquences du bloc). Pour les suivants, le gain est négligeable car, du fait de la quantification, leur valeur d'origine est nulle. Nous constatons que le meilleur gain pour les trois contextes de codage est obtenu pour $I_{DCT} = [c_{10}]$, surtout pour les bas-débits ($1,68\%$ de gain). En effet, dans ce cas seuls les coefficients correspondant aux basses fréquences sont transmis, les autres étant quantifiés à une valeur nulle. De ce fait, la prédiction de l'un de ces coefficients entraîne un gain en débit plus important par rapport à la quantité d'information totale transmise. Pour ces mêmes raisons, nous constatons que la tendance s'inverse lorsque la prédiction porte sur les coefficients correspondant à des plus hautes fréquences.

Dans un deuxième temps, nous souhaitons savoir si la réduction du débit est plus importante avec plusieurs coefficients DCT prédits. Pour cela, nous fixons $I_{DCT} = [c_{10}, c_{ij}]$, avec $c_{ij} \neq c_{10}$. Les résultats sont illustrés sur la figure 6.8. Les gains observés sont nettement améliorés par rapport à 6.7, avec en moyenne un gain de $2,51\%$ pour $I_{DCT} = [c_{10}, c_{01}]$. Ces chiffres correspondent approximativement à la somme des gains obtenus avec $I_{DCT} = [c_{10}]$ et $I_{DCT} = [c_{01}]$. Nous en concluons que la prédiction de l'un de ces coefficients n'affecte pas la prédiction de l'autre. Là encore, les meilleurs gains sont atteints dans un contexte de codage à bas-débit.

Par la suite, nous ne constatons pas de réduction de débit supplémentaire lorsque nous ajoutons d'autres coefficients prédits à l'ensemble $I_{DCT} = [c_{10}, c_{01}]$. Nous en concluons que le meilleur com-

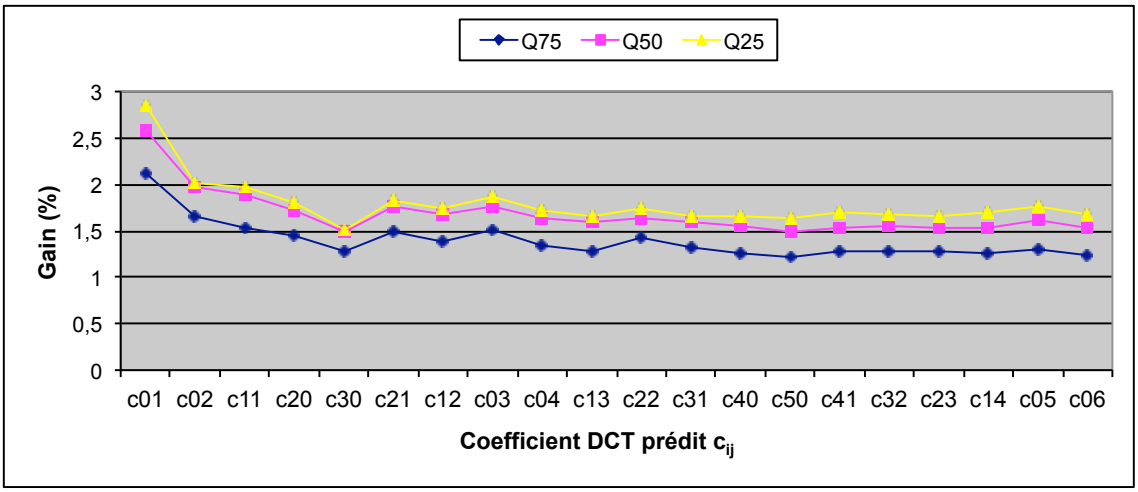


FIGURE 6.8 – Gain moyen en débit (réf : JPEG) en fonction d'un coefficient c_{ij} prédit, avec $I_{DCT} = [c_{10}, c_{ij}]$, et d'une qualité de codage $Q \in [25, 50, 75]$.

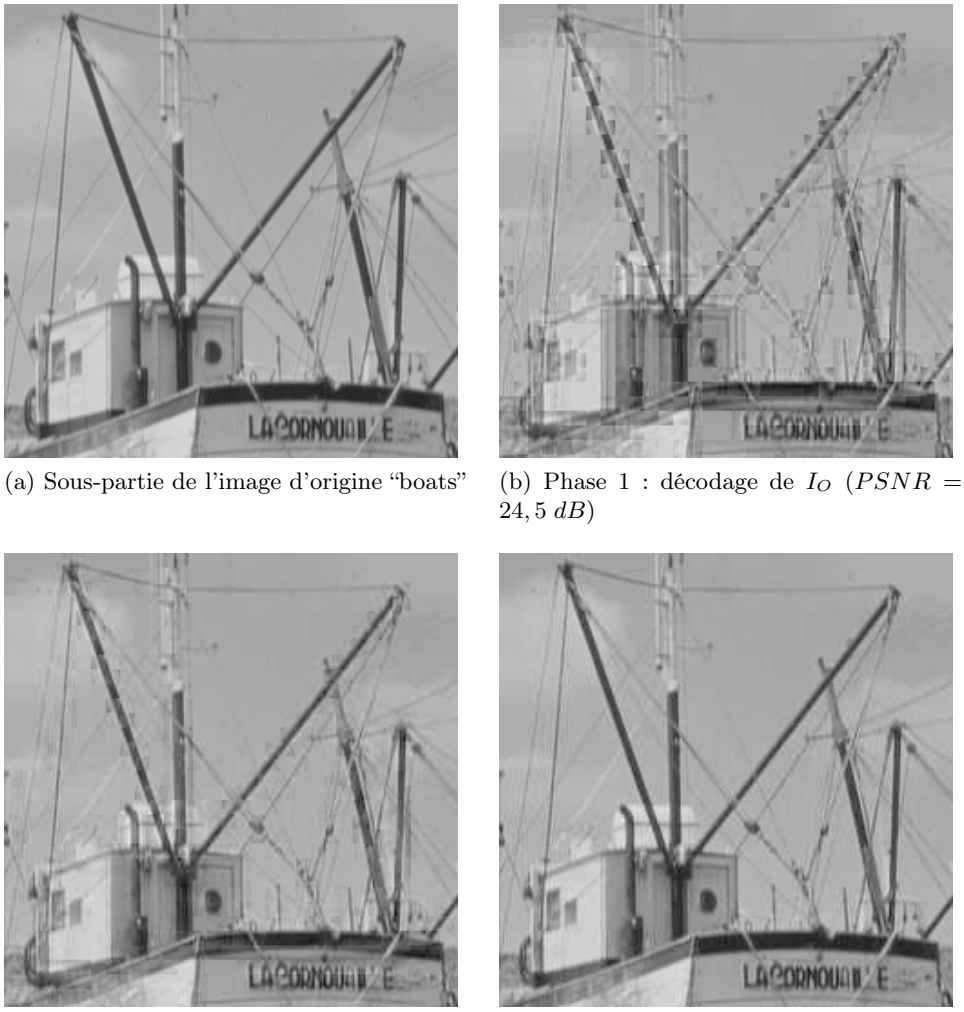
	$Q75$		$Q50$		$Q25$	
image	débit (bpp)	Δ (%)	débit (bpp)	Δ (%)	débit (bpp)	Δ (%)
barbara	1,32	2,24	0,89	2,88	0,56	2,88
boats	0,95	2,19	0,63	2,93	0,39	3,44
bridge	1,87	1,47	1,21	1,88	0,74	2,27
couple	1,29	2,02	0,85	2,3	0,52	2,62
crowd	1,22	3,27	0,84	3,88	0,54	4,14
dollar	2,14	0,59	1,41	0,63	0,87	0,73
girlface	0,81	3,82	0,5	4,77	0,3	5,24
kiel	1,48	1,01	0,97	1,27	0,59	1,44

TABLE 6.1 – Méthode de prédiction des coefficients DCT dans JPEG : résultats par image et par débit (réf : JPEG). Le débit obtenu est exprimé en bits par pixel. Le gain par rapport à JPEG est exprimé en pourcentage.

promis entre les coefficients transmis et les coefficients prédits est atteint pour cet ensemble I_{DCT}. Nous constatons donc que la méthode fonctionne de façon optimale lorsque ce sont les deux coefficients c_{10} et c_{01} de plus forte énergie et correspondant aux basses fréquences du bloc qui sont prédits, quelque soit le débit souhaité. Sur l'ensemble des huit images testées, le moins bon résultat est enregistré pour l'image "dollar" à $Q = 75$, avec $0,59\%$ de réduction de débit. A l'inverse, le maximum est atteint avec l'image "girlface" pour $Q = 25$ avec $5,24\%$ de réduction de débit. Le tableau 6.1 résume les résultats pour l'ensemble des images. La méthode possède les meilleurs rendements pour les images avec peu de textures et composées en grande partie de dégradés. Ce constat s'explique par le fait que le modèle de prédiction part de l'hypothèse que l'image doit être restaurée de façon à être la plus lisse possible.

La figure 6.9 illustre les différentes phases du décodage des images selon la méthode de prédiction des coefficients DCT étudiée dans cette section. L'image "boats" est encodée avec les paramètres $Q = 50$ et $I_{DCT} = [c_{10}, c_{01}]$ pour un débit de $0,63$ bpp. En particulier, la sous-figure 6.9a représente une sous-partie de l'image d'origine. La première phase de décodage, illustrée sur la sous-figure 6.9b,

restitue l'image sans prendre en considération les coefficients DCT prédits appartenant à I_{DCT}. La seconde phase de décodage (6.9c) va consister à prédire les coefficients et $[c_{10}, c_{01}] \in I_{DCT}$ à partir de 6.9b. Enfin, la dernière phase (6.9d) restitue l'image finalement décodée en prenant en compte les coefficients prédits et l'erreur de prédiction transmise.



(a) Sous-partie de l'image d'origine "boats"

(b) Phase 1 : décodage de I_O ($PSNR = 24,5\ dB$)

(c) Phase 2 : prédiction de I_{DCT} ($PSNR = 28,15\ dB$)

(d) Phase 3 : ajout de l'erreur de prédiction pour I_{DCT} ($PSNR = 34,68\ dB$)

FIGURE 6.9 – Illustration des phases de décodage de notre algorithme de compression basé sur la méthode de prédiction des coefficients DCT.

Cependant, la réduction de débit reste modeste pour un codeur de type JPEG (autour de $2,5\%$ en moyenne). Nous allons voir dans la section suivante comment adapter cette méthode dans un contexte de codage vidéo, avec le double objectif de prendre en compte les spécificités propres au codage prédictif des standards vidéo et d'améliorer la méthode proposée pour obtenir des gains intéressants.

6.5 Codage vidéo par prédiction intra-bloc des coefficients DCT

L'aboutissement de ce chapitre de contribution amène à intégrer les éléments précédemment introduits dans un nouveau schéma de codage vidéo de type H.264/AVC. Dans un premier temps, il est nécessaire d'identifier les spécificités de H.264/AVC afin d'adapter le modèle théorique de restauration vu en 6.2. Ensuite, nous introduisons les différents choix de mise en oeuvre qui nous semblent les plus judicieux dans ce contexte d'utilisation et qui se différencient par rapport à ce qui a été fait dans 6.4. Ensuite, nous présenterons le schéma de codage vidéo qui en ressort. Enfin, le détail des expérimentations et les résultats sont exposés.

6.5.1 Adaptation du modèle théorique aux contraintes de codage vidéo de type H.264/AVC

Par rapport au schéma de codage d'image fixe de type JPEG présenté en 6.4, le schéma de codage étudié dans cette section repose sur une étape de prédiction intra-image ou inter-image supplémentaire (voir 2.3.2). Il est à noter que nous différencions les concepts de prédiction propres à H.264/AVC (prédiction intra-image ou inter-image des pixels) de notre méthode de prédiction des coefficients DCT (que nous qualifions de prédiction intra-bloc). Par rapport au modèle de restauration présenté dans 6.2, un bloc d'image u_B est défini par la somme de deux composantes : un bloc prédicteur de type intra ou inter image, p, et un résidu, r, transformé dans le domaine DCT (p et r sont restreints au bloc B pour plus de lisibilité). Nous avons donc la relation suivante :

$$u_{B\vec{z}} = p_{\vec{z}} + r_{\vec{k}}\phi_{\vec{z},\vec{k}} \tag{6.19}$$

où $\vec{k} = \begin{pmatrix} i \\ j \end{pmatrix}$ correspond à la position du coefficient DCT dans le bloc résiduel r, et $\vec{z} = \begin{pmatrix} x \\ y \end{pmatrix}$ la position du pixel dans les blocs u_B et p, et $\phi_{\vec{z},\vec{k}}$ le noyau de la transformée DCT modifiée, adoptée dans H.264/AVC ([118]).

L'expression de la variation totale de u_B pour tous les blocs B de l'image est donnée par :

$$\left\| u_{B\vec{z}}\left(p_{\vec{z}}, r_{\vec{k}}\right)\right\|_{TV} = \int_{\Omega} \left|\nabla_{\vec{z}} u_{B\vec{z}}\left(p_{\vec{z}}, r_{\vec{k}}\right)\right| d\vec{z} \tag{6.20}$$

où Ω est défini dans $\mathbb{R}^2$.

Notre méthode de prédiction va s'appliquer sur les coefficients transformés du résidu r. Pour cela, certains coefficients de r, ceux appartenant à l'ensemble I_{DCT}, sont prédits au lieu d'être directement transmis. L'erreur de notre prédiction intra-bloc est codée pour permettre la reconstruction du bloc u_B au décodage. Notre modèle consiste, de façon analogue à ce que nous avons vu précédemment dans 6.4, à prédire I_{DCT} en minimisant la variation totale de l'image. Afin d'arriver à ce but, nous exprimons la dérivée partielle de la variation totale de $u_{B\vec{z}}$ en fonction du résidu $r_{\vec{k}}$ pour tous les blocs B de l'image :

$$\begin{aligned}\frac{\partial TV(u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}}))}{\partial r_{\vec{k}}} &= \int_\Omega \frac{\partial |\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})|}{\partial r_{\vec{k}}} \mathrm{d}\vec{z} \\ &= \int_\Omega \frac{\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})}{|\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})|} \cdot \frac{\partial \nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})}{\partial r_{\vec{k}}} \mathrm{d}\vec{z} \\ &= \int_\Omega \frac{\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})}{|\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})|} \cdot \nabla_{\vec{z}} \frac{\partial u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})}{\partial r_{\vec{k}}} \mathrm{d}\vec{z} \end{aligned} \tag{6.21}$$

Avec par définition l'équation :

$$\frac{\partial u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})}{\partial r_{\vec{k}}} = \phi_{\vec{z},\vec{k}}, \tag{6.22}$$

Ce qui permet d'écrire :

$$\frac{\partial TV(u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}}))}{\partial r_{\vec{k}}} = \int_\Omega \frac{\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})}{|\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})|} \cdot \nabla_{\vec{z}} \phi_{\vec{z},\vec{k}} \mathrm{d}\vec{z} \tag{6.23}$$

En utilisant une intégration par parties, nous avons :

$$\begin{aligned}\int_\Omega \frac{\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})}{|\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})|} \cdot \nabla_{\vec{z}} \phi_{\vec{z},\vec{k}} \mathrm{d}\vec{z} = &\left[\nabla_{\vec{z}} \cdot \left(\frac{\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})}{|\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})|} \phi_{\vec{z},\vec{k}} \right) \right]_0^\Omega \\ &- \int_\Omega \left(\nabla_{\vec{z}} \cdot \frac{\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})}{|\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})|} \right) \phi_{\vec{z},\vec{k}} \mathrm{d}\vec{z} \end{aligned} \tag{6.24}$$

A cause du partitionnement et du traitement par bloc, le noyau DCT $\phi_{\vec{z},\vec{k}}$ est nul en dehors du bloc u_B, ce qui se traduit par :

$$\left[\nabla_{\vec{z}} \cdot \left(\frac{\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})}{|\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})|} \phi_{\vec{z},\vec{k}} \right) \right]_0^\Omega = 0 \tag{6.25}$$

Au final, nous retrouvons l'équation de la dérivée partielle définie par l'expression de la courbure de l'image transposée dans le domaine DCT :

$$\frac{\partial TV(u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}}))}{\partial r_{\vec{k}}} = - \int_\Omega \left(\nabla_{\vec{z}} \cdot \frac{\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})}{|\nabla_{\vec{z}} u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})|} \right) \phi_{\vec{z},\vec{k}} \mathrm{d}\vec{z} \tag{6.26}$$

En revenant au problème d'origine, le modèle consiste à régulariser l'image par minimisation de la variation totale en modifiant certains coefficients DCT définis dans I_{DCT} du résidu r :

$$\min_{r_{\vec{k}}, \vec{k} \in \mathcal{I}_{DCT}} TV(u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}})) \tag{6.27}$$

Une solution stable à cette minimisation est donnée par l'équation d'Euler-Lagrange associée :

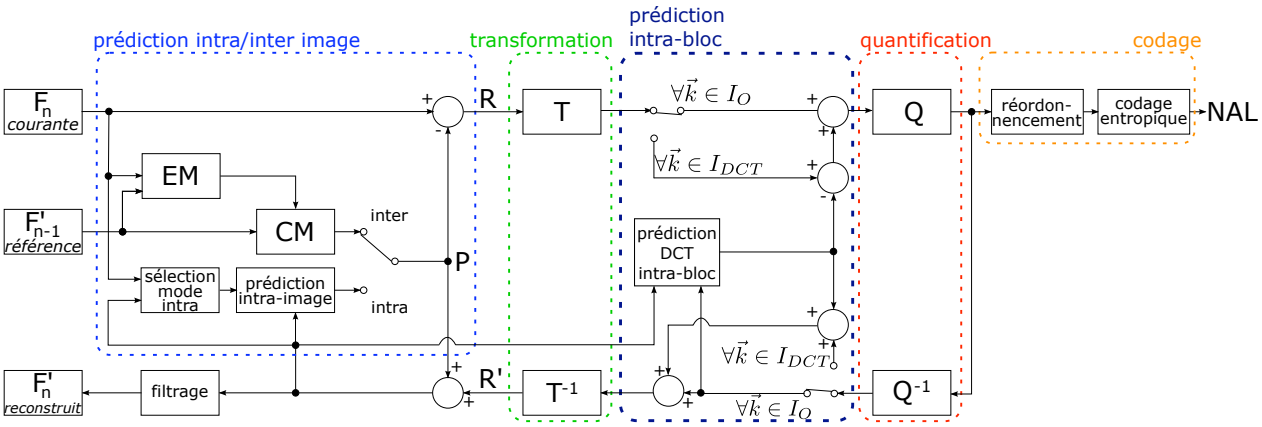


FIGURE 6.10 – Schéma de codage vidéo basé sur H.264/AVC avec l'étape supplémentaire de prédiction intra-bloc des coefficients DCT (se référer à la Fig. 2.5 pour le schéma de codage H.264/AVC).

$$\frac{\partial TV(u_{\vec{z}B}(p_{\vec{z}}, r_{\vec{k}}))}{\partial r_{\vec{k}}} = 0 \tag{6.28}$$

Les conclusions sont identiques à celles de la section 6.4. Au final, l'ajout d'une étape de prédiction intra/inter-image n'affecte pas notre modèle de prédiction intra-bloc. La mise en pratique d'un algorithme de restauration, utilisé en tant qu'étape de prédiction dans un schéma de compression, est donc analogue à 6.1, en prenant soin d'additionner les deux composantes p et r pour extraire la courbure de l'image.

6.5.2 Schéma de codage vidéo proposé

Sur la base du modèle précédemment introduit, une nouvelle méthode de prédiction est intégrée dans le schéma de codage H.264/AVC. Des quatre grandes étapes de codage classiques, à savoir prédiction intra/inter image, transformation, quantification du résidu et codage entropique (voir figure 2.5), nous ajoutons l'étape de prédiction intra-bloc des coefficients DCT comme illustré sur la figure 6.10. La prédiction des coefficients DCT appartenant à I_{DCT} est effectuée en utilisant deux types de données en entrée : les coefficients $c_{ij} \in I_O$ (les coefficients DCT non prédits) et les blocs adjacents précédemment encodés et reconstruits, ceci afin de calculer la courbure de l'image sur une portion plus grande que le bloc lui-même. En sortie du module de prédiction intra-bloc, nous obtenons une estimation du coefficient d'origine qui va nous permettre de calculer le résidu du côté de l'encodeur, soit l'erreur de prédiction intra-bloc, par une simple opération de soustraction avec le coefficient d'origine. L'association des deux ensembles de coefficients I_O et I_{DCT} est ensuite effectuée avant l'étape de quantification. Au décodeur, le résidu déquantifié est ajouté à la prédiction intra-bloc pour les coefficients de I_{DCT} afin de reconstruire le bloc avec les coefficients d'origine. Nous pouvons noter que si I_{DCT} est vide, alors nous exploitons un schéma de codage vidéo classique respectant le standard H.264/AVC.

Comme pour le codage d'image fixe vu en 6.4, l'objectif de cette étape supplémentaire de prédiction DCT intra-bloc est de réduire la quantité d'information transmise en réduisant l'entropie du résidu de prédiction par rapport aux coefficients DCT d'origine.

L'un des avantages du schéma de codage présenté est qu'à ce stade, la transmission d'information de signalisation supplémentaire n'est pas requise puisque l'étape de prédiction intra-bloc s'applique sur tous les blocs sans exception.

6.5.3 Spécificités du codage vidéo et choix d'implémentation

Dans le standard H.264/AVC FRext, la transformation DCT est appliquée sur des blocs de taille 4×4 et 8×8 pixels. Nous appliquons la méthode de prédiction intra-bloc pour ces deux tailles de blocs. A noter également que la transformation de type DCT utilisée dans H.264/AVC (2.3.2.3) est une approximation de la DCT telle qu'elle est définie dans 6.1.2, ceci afin de faciliter l'implémentation par des entiers et des opérateurs simples de décalage de bits. La transformation étant réversible, nous utiliserons le même type de transformée et transformée inverse modifiées dans l'implémentation de notre méthode de prédiction intra-bloc des coefficients DCT.

Dans un schéma classique de codage vidéo (voir Fig. 2.5), l'étape de prédiction intra ou inter image a l'avantage de réduire l'énergie du résidu par rapport à l'information initiale, tout en nécessitant un coût de signalisation restreint. Dés lors, le gain de débit en intégrant notre méthode de prédiction intra-bloc sera plus faible qu'en 6.4, d'autant plus que les précédentes expérimentations ont montré que les gains de débit maximum étaient atteints lorsque les coefficients DCT de plus forte énergie étaient prédits (les coefficient c_{01} et c_{10} du bloc). De ce constat, un nouveau mécanisme de prédiction itératif des coefficients DCT est adopté dans le but d'optimiser les performances en compression. L'idée consiste à prédire les coefficients appartenant à I_{DCT} successivement au lieu d'une prédiction simultanée. Lorsqu'un coefficient donné $\vec{k}$ est prédit puis codé/décodé, il est alors ajouté à l'ensemble I_O pour être exploité ultérieurement, c'est-à-dire pour prédire les coefficients suivants de l'ensemble I_{DCT} du même bloc. Au fur et à mesure du processus, l'ensemble des coefficients de I_O s'accroît, rendant ainsi la prédiction des coefficients restants de I_{DCT} plus efficace. Intuitivement, l'ordre de prédiction des coefficients de I_{DCT} est dicté par le parcours *zigzag* des blocs, c'est-à-dire que les coefficients DCT correspondant aux plus basses fréquences du signal sont prédits en priorité. L'algorithme résultant est présenté en 6.2. A l'étape 4e de l'algorithme, nous obtenons $\beta_{\vec{c}}^{n+1}$, la prédiction du coefficient $\vec{c} \in I_{DCT}$ courant.

En ce qui concerne le choix de l'ensemble I_{DCT} des coefficients à prédire, le même principe que pour le codage d'image fixe évoqué dans 6.4.1 est utilisé. Il consiste à prédéfinir un masque au codeur et au décodeur. Ce mécanisme permet avant tout de ne pas nécessiter de codage d'information supplémentaire afin de spécifier I_{DCT}.

Cependant, dans le contexte de codage intra-image, nous avons un *a priori* statistique sur la répartition de l'énergie des coefficients DCT résiduels d'un bloc en fonction de son mode de prédiction intra-image. Le lien qui unit le mode de prédiction intra-image et la répartition de l'énergie du résidu a, entre autre, été étudié dans [156]. L'une des applications tirée de ces travaux consiste à adapter l'ordre de parcours des coefficients DCT dans un bloc [59, 84, 77]. Initialement, dans H.264/AVC, l'ordre de parcours utilisé est le parcours en *zigzag* [118] qui traite les coefficients DCT de la plus basse fréquence à la plus haute. Cependant, l'ajout d'une méthode de prédiction intra-image peut potentiellement changer l'ordre optimal de parcours des coefficients DCT. L'objectif des ordres de parcours alternatifs est de concentrer les coefficients nuls à la fin du parcours, et ce afin que le codage entropique puisse en tirer profit.

Dans notre méthode de prédiction intra-bloc, nous réutilisons ces concepts afin de prédire les coefficients qui ont statistiquement la plus grande amplitude en fonction du mode de prédiction intra-

Algorithme 6.2 Méthode de prédiction intra-bloc utilisée dans un contexte de codage vidéo.

1. Soit p le prédicteur intra ou inter image du bloc courant u_B
2. Soit $r = DCT\left(u_B - p\right)$ le résidu transformé de p, et $\widehat{r}$ le résidu r décodé pour tout coefficient $\vec{k} \in I_O$
3. Soit K le nombre maximum d'itérations
4. **Tant que** $I_{DCT} \neq \emptyset$ **, faire**
   (a) Soit $\beta_{\vec{k}} = \begin{cases} \widehat{r}_{\vec{k}} & \text{si} \quad \vec{k} \in I_O \\ 0 & \text{sinon} \end{cases}$
   (b) Soit $\vec{c} \in I_{DCT}$ l'indice du coefficient à prédire, avec $\vec{c}$ correspondant au premier coefficient DCT du bloc selon parcours zig-zag
   (c) Soit $n = 0$ l'itération courante et γ_n le pas de descente de gradient pour la prédiction du coefficient $\vec{\beta}$
   (d) **Tant que** $n < K$**, faire**
      i. Soit $s = DCT\left(curv\left(p + IDCT\left(\beta^n\right)\right)\right)$
      ii. $\beta_{\vec{c}}^{n+1} = \beta_{\vec{c}}^{n} + \gamma_n s_{\vec{c}}$
      iii. Calculer $n = n + 1$
   (e) **Fin de la boucle**
   (f) Coder/décoder le résidu de la prédiction intra-bloc $r_{\vec{c}} - \beta_{\vec{c}}^{n+1}$ du coefficient $\vec{c}$ pour obtenir $\widehat{r_{\vec{c}} - \beta_{\vec{c}}^{n+1}}$, et reconstruire $\widehat{r}_{\vec{c}} = \beta_{\vec{c}}^{n+1} + \widehat{r_{\vec{c}} - \beta_{\vec{c}}^{n+1}}$
   (g) Mettre à jour I_{DCT} et I_O pour que $I_{DCT} = I_{DCT}/\vec{c}$, $I_O = I_O \cup \vec{c}$
5. **Fin de la boucle**

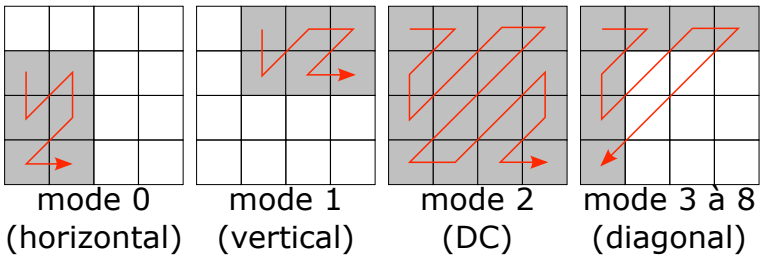


FIGURE 6.11 – Position (en gris) des coefficients prédits de l'ensemble I_{DCT} pour les blocs de taille 4×4. En rouge figure l'ordre de prédiction des coefficients.

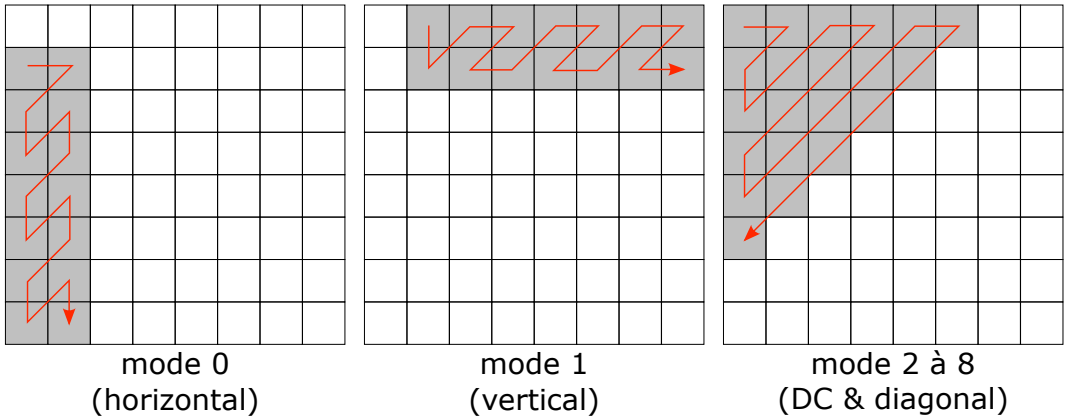


FIGURE 6.12 – Position (en gris) des coefficients prédits de l'ensemble I_{DCT} pour les blocs de taille 8×8. En rouge figure l'ordre de prédiction des coefficients.

image, car nous avons vu précédemment dans 6.4 cette approche garantissait systématiquement les meilleurs résultats. Nous avons décliné I_{DCT} en fonction des neufs modes de prédiction intra-image de façon expérimentale, les coefficients prédits sont exposés en gris sur les figures 6.11 et 6.12 pour les blocs de taille 4×4 et 8×8 coefficients respectivement. Le parcours itératif des coefficients prédits est illustré par la flèche rouge, il reprend le parcours en *zigzag* classique appliqué sur l'ensemble des coefficients de I_{DCT}. Intuitivement, lorsque la prédiction intra-image d'un bloc B correspond à une interpolation avec les pixels situés au dessus de B (prédiction verticale), l'énergie du bloc transformé résiduel sera majoritaire sur les coefficients DCT correspondant aux oscillations horizontales.

6.5.4 Expérimentations et résultats

Nous avons implémenté l'algorithme 6.2 de prédiction intra-bloc des coefficients DCT dans la plate-forme logicielle de référence JSVM 9.7 [125] compatible avec H.264/AVC (c'est-à-dire en désactivant la partie destinée au codage vidéo scalable). Nous activons le codeur entropique CABAC, ainsi que les transformées DCT pour les blocs 4×4 et 8×8 pixels. Dans ces expérimentations, toutes les images sont codées en mode intra-image. La prédiction DCT intra-bloc est activée pour tous les blocs de luminance de taille 4×4 et 8×8, en reprenant les masques donnés par les figures 6.11 et 6.12, qui sont définis en fonction du mode de prédiction intra-image utilisé. Pour l'algorithme de descente de gradient utilisé pour prédire chaque coefficient DCT, le nombre d'itérations est fixé à 100 ($K = 100$ dans l'algorithme 6.2), avec un pas fixe ($\gamma_n = c^{ste}$). L'optimisation de ces paramètres,

Séquence	Gain de débit (%)
CIF	
Foreman	1.88
Mobile	2.06
Paris	2.20
Tempete	2.40
MOYENNE CIF	**2.13**
720p	
BigShip	1.40
City	1.77
Night	2.05
OldTownCross	**2.58**
Raven	2.02
ShuttleStart	2.51
MOYENNE 720p	**2.05**

TABLE 6.2 – Pourcentage de gain de débit exprimé par la métrique de Bjontegaard et par rapport à la référence H.264/AVC.

voire le recours à d'autres algorithmes plus adaptés, permettraient probablement de converger vers une solution plus rapidement. Ceci étant, l'objectif est en priorité de démontrer que des gains en débit peuvent être obtenus avec ce type de méthode.

La métrique de Bjontegaard [25] est utilisée pour mesurer les résultats en pourcentage de notre nouvelle méthode de codage par rapport à H.264/AVC, en considérant quatre couples qualité/débit différents définis, dans H.264/AVC par $QP = 22,\ 27,\ 32,\ 37$ (voir [118] pour la signification du paramètre QP qui définit le pas de quantification). La qualité mesurée est objective est s'obtient par une mesure de PSNR. Les gains en débit obtenus sont exprimés en pourcentage pour une qualité équivalente.

L'un des avantages principaux de notre méthode de prédiction intra-bloc des coefficients DCT est qu'elle ne requiert pas de transmission d'information additionnelle de signalisation, puisque la méthode s'applique sur tous les blocs de l'image et que l'ensemble I_{DCT} est prédéfini au codeur comme au décodeur. Le résidu de prédiction intra-bloc d'un coefficient DCT est transmis en lieu et place du coefficient DCT d'origine, après quantification et codage selon les mêmes paramètres.

Nous avons expérimenté notre méthode de codage sur des séquences vidéo de résolution CIF (352×288 pixels) et 720p (1280×720 pixels). Les résultats sont présentés sur le tableau 6.2, où seules les 49 premières images de la séquences sont encodées. Les gains observés sont similaires entre les deux formats de séquence CIF et 720p, avec une moyenne d'un peu plus de 2% de gain de débit par rapport à la référence H.264/AVC. Nous constatons cependant une légère disparité des résultats en fonction des séquences, qui s'explique de la même façon que pour le codage d'image fixe vu en 6.4. En effet, notre méthode de prédiction des coefficients DCT fonctionne sur l'hypothèse que l'image doit être régularisée de façon à être la plus lisse. La méthode donne de meilleurs résultats lorsqu'elle l'est effectivement (exemple de la séquence *OldTownCross*).

Sur la figure 6.13, nous illustrons en pourcentage le gain obtenu en fonction du débit pour les séquences CIF. Il est intéressant de noter qu'aux extrémités basses de débit, les performances ont tendance à diminuer pour toutes les séquences. Ce constat est lié au fait que, à bas-débit, la majorité

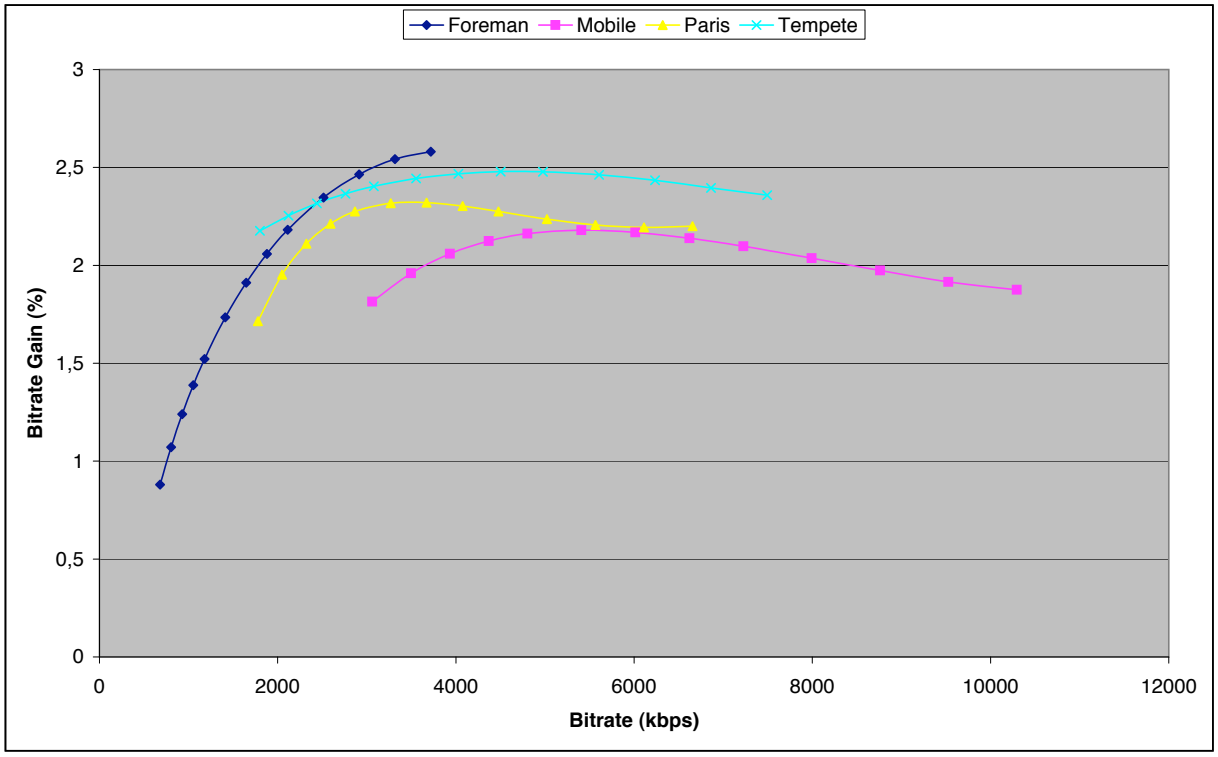


FIGURE 6.13 – Pourcentage de gain en compression obtenu en fonction du débit pour quatre séquences CIF

des coefficients appartenant à l'ensemble I_O, sur lequel s'appuie le procédé de prédiction intra-bloc, sont nuls après quantification. Comme nous l'avons dans la partie 6.3.1, moins il y a d'information disponible pour prédire I_{DCT}, moins la restauration sera efficace.

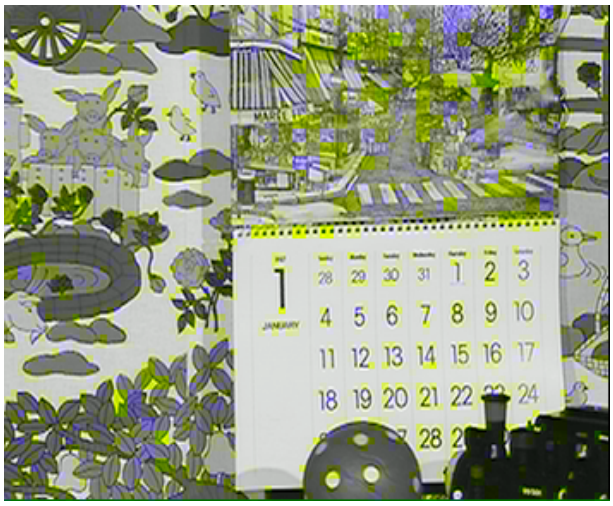
Enfin, la figure 6.14 montre les gains obtenus par bloc sur la première image de la séquence *Foreman* à différents débits. Nous remarquons alors deux choses. Premièrement, c'est à haut-débit qu'il y a le plus de disparité entre les coûts de codage des blocs par rapport à la référence H.264/AVC. Le pas de quantification étant plus faible dans ce cas, il y a d'autant plus de coefficients non nuls à coder, donc de coefficients sur lesquels notre méthode de prédiction peut donner des résultats intéressants. Deuxièmement, pour certains blocs, la méthode que nous avons élaborée engendre un surplus d'information à coder, donc le résidu de prédiction intra-bloc a une plus forte amplitude que le coefficient d'origine. Il semble cependant que certains blocs voisins, aux aspects très similaires, donnent des résultats finalement très différents en terme de coût de codage. Nous en concluons qu'il n'y a pas de région "type" d'image (contour, texture, ...) pour lesquels notre méthode possède un meilleur rendement, si ce n'est pour les régions totalement plates où aucune différence de coût de codage n'est notable. C'est d'autant plus flagrant avec l'image *Mobile* illustrée sur la figure 6.15.

(a) $QP = 22$ (haut-débit) (b) $QP = 27$

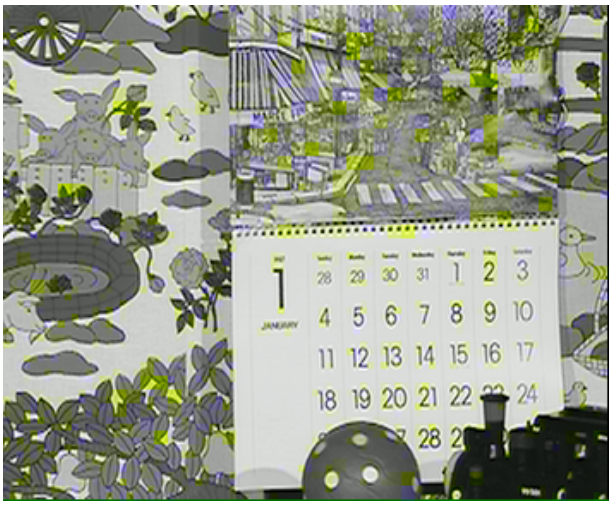
(c) $QP = 32$ (d) $QP = 37$ (bas-débit)

FIGURE 6.14 – Résultats obtenus sur la première image de la séquence *Foreman*. En vert : les blocs où la prédiction intra-bloc réduit le nombre de symboles codés par rapport à H.264/AVC. En bleu : lorsque le nombre de symboles codés augmente. Plus l'intensité chromatique est élevée, plus la différence est importante.

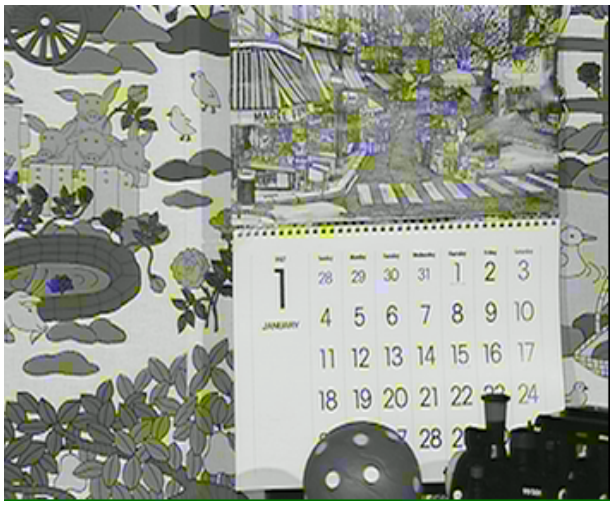
(a) $QP = 22$ (haut-débit)



(b) $QP = 27$



(c) $QP = 32$



(d) $QP = 37$ (bas-débit)

FIGURE 6.15 – Résultats obtenus sur la première image de la séquence *Mobile*. En vert : les blocs où la prédiction intra-bloc réduit le nombre de symboles codés par rapport à H.264/AVC. En bleu : lorsque le nombre de symboles codés augmente. Plus l'intensité chromatique est élevée, plus la différence est importante.

6.6 Conclusion

Nous avons présenté un modèle de régularisation d'image, basé sur la variation totale, que nous avons ensuite décliné pour obtenir un algorithme de restauration de coefficients DCT. Notre modèle est sujet à deux contraintes : la minimisation de la variation totale d'un côté, qui produit une image lisse mais dont les contours restent marqués, et la restauration des coefficients DCT dégradés uniquement, ce qui permet de ne pas modifier l'information d'origine disponible. Nous avons ensuite adapté l'algorithme de restauration pour l'intégrer dans un schéma de codage d'image fixe de type JPEG. Ainsi une nouvelle étape de prédiction de certains coefficients DCT permet de réduire l'énergie du résidu de prédiction par rapport aux coefficients d'origine, ce qui permet de réduire le coût de codage d'environ $2,5\%$ par rapport à JPEG. Enfin, ce modèle de prédiction des coefficients DCT a été intégré dans un codeur vidéo de type H.264/AVC en respectant les contraintes de ce type de schéma, notamment l'étape de prédiction intra et inter image. L'implémentation s'est focalisée sur les images de type I, c'est-à-dire codées uniquement selon un procédé de prédiction intra-image. Les résultats obtenus ont montré un gain de débit de 2% en moyenne par rapport à H.264/AVC, avec un maximum à $2,52\%$ pour l'image *OldTownCross*.

Cependant, notre méthode de prédiction intra-bloc est difficilement exploitable sous cette forme du fait de sa complexité calculatoire. Pour illustrer ces propos, prenons le cas où un bloc de taille 4×4 subit une prédiction intra-image avec le mode DC. De ce fait, et en référence à la figure 6.11, une prédiction de type intra-bloc sera appliquée sur les seize coefficients du blocs tour à tour. Puisque nous avons paramétré le nombre d'itérations de l'algorithme de descente de gradient à 100 durant les expérimentations, et que pour chaque itération il est nécessaire d'effectuer une DCT et une DCT inverse, nous avons en tout 1600 opérations de DCT et DCT inverses pour prédire un bloc 4×4 au codeur comme au décodeur, au lieu d'une seule dans le cas de H.264/AVC. Il existe cependant plusieurs façons de réduire la complexité. Tout d'abord, en ne traitant qu'un seul coefficient DCT à la fois, il n'est pas nécessaire d'effectuer une DCT et une DCT inverse complète à chaque itération pour l'ensemble du bloc. De plus, nous utilisons un algorithme de descente de gradient générique avec un nombre d'itérations fixé à 100 et un pas de descente de gradient fixe. En ajustant ces deux paramètres et en ajoutant une contrainte de sortie de l'algorithme lorsqu'un état stable est atteint, nous pourrions considérablement réduire le nombre d'itérations nécessaires et diminuer la complexité de la méthode. Encore une fois, l'objectif était avant tout de montrer que ce type de méthode peut donner des résultats intéressants, sachant que ces expérimentations ont été réalisés sans prendre en considération l'aspect de la complexité calculatoire.

Le mécanisme de prédiction intra-bloc que nous avons élaboré peut s'étendre aux blocs prédits temporellement (prédiction inter-image), avec toutefois quelques remarques. Premièrement, le résidu de prédiction inter-image étant globalement plus faible que pour la prédiction intra-image (la prédiction étant plus efficace), il y aura majoritairement plus de coefficients DCT nuls dans ce cas. De ce fait, la prédiction intra-bloc aura un rendement naturellement plus faible. Deuxièmement, nous nous sommes appuyés sur la répartition statistique de l'énergie d'un bloc transformé en fonction du mode de prédiction intra-image pour optimiser l'ensemble I_{DCT}, l'objectif étant de prédire les coefficients ayant la plus forte amplitude. Or, la prédiction inter-image ne permet pas, à première vue, d'avoir d'*a priori* sur la répartition de l'énergie dans le bloc, donc d'utiliser ce type de mécanisme pour optimiser les performances.

Enfin, l'approche de prédiction intra-bloc présentée peut s'avérer intéressante dans un schéma de

codage vidéo scalable (SVC [1], voir [126]). La scalabilité offre la possibilité de pouvoir représenter un signal à différents niveaux d'information. Cette hiérarchisation repose sur une "couche" de base, qui concentre l'information pertinente, et sur une multitude de couches de réhaussement qui agrémentent l'information initiale. Sur les différents types de scalabilité, nous nous intéressons notamment à la scalabilité en qualité. Dans ce cas, les coefficients transformés sont successivement affinés au fur et à mesure des couches de réhaussement, ce qui a pour effet d'améliorer la qualité de l'image décodée. L'idée que nous avançons consiste à prédire le raffinement des coefficients à partir de la couche inférieure, selon la méthode que nous avons exposée. Ainsi, l'erreur de prédiction des couches de qualité étant globalement plus faible que l'information d'origine, ce mécanisme pourrait permettre de réduire les coûts de codage.

1. *Scalable Video Coding*

Quatrième partie

Conclusion et perspectives

Les travaux de cette thèse s'inscrivent dans le cadre du codage vidéo. Dans ce domaine, nous avons vu qu'il était nécessaire de formaliser de nouveaux standards de codage, notamment pour faire face à l'augmentation de la demande des opérateurs de télécommunication. Pour cela, la nouvelle génération de standard de codage vidéo doit permettre d'augmenter significativement le ratio de compression de l'ordre de 50% par rapport au standard actuel, tout en conservant une complexité de codage et de décodage acceptable.

Le travail effectué s'est réparti sur deux axes de recherches distincts, s'articulant néanmoins autour du même objectif commun défini par le sujet de la thèse, à savoir l'amélioration des performances en compression dans un contexte de codage intra-image.

Le premier axe de recherche a trouvé sa source dans le domaine du traitement d'image et plus précisément sur les méthodes de synthèse de texture. Ces méthodes sont utilisées pour reproduire automatiquement et à l'infini une texture de façon à conserver ses caractéristiques perceptuelles intrinsèques. Nous nous sommes basés sur cette approche pour concevoir un outil de prédiction spatiale de texture que nous avons intégré dans un schéma de codage de type H.264/AVC. En effet, dans les schémas de codage actuels, nous constatons une baisse significative d'efficacité en compression lorsque les séquences vidéo sont constituées de textures rigides complexes. Le mécanisme de prédiction intra-image des codeurs de type H.264/AVC n'est tout simplement pas adapté à ce type de modélisation. Par le fait qu'il repose sur une interpolation des pixels adjacents du bloc, la prédiction intra-image produit un effet de lissage non représentatif d'une texture rigide. Notre algorithme permet de combler cette faiblesse en "recopiant" des échantillons de texture préalablement codés en effectuant une analyse du voisinage.

Notre travail se distingue des autres méthodes de prédiction de texture dédiées au codage vidéo, méthodes dites par *template matching*. En effet, il nous a semblé judicieux de prendre en considération les aspects d'estimation objective de la qualité, utilisés dans les schémas de codage vidéo, incompatibles avec les méthodes de synthèse de texture "pures" basées sur des critères perceptuels. Pour cela, nous utilisons un ensemble de prédicteurs de texture triés par vraisemblance au lieu de ne privilégier qu'un seul représentant. Ce mécanisme nécessite néanmoins de coder une information supplémentaire pour réaliser l'opération inverse de décodage. Les expérimentations ont permis de prouver que notre approche offre une efficacité débit-distorsion supérieure par rapport à l'état de l'art. De plus, les tests ont mis en évidence le fait que notre outil de prédiction s'étend en dehors du cadre de la modélisation de textures complexes. Il se trouve en effet que notre algorithme peut se montrer plus rentable pour prédire d'autres types de contenus ; notamment des isophotes rectilignes dont l'orientation ne correspond pas à l'un des angles d'interpolation des méthodes de prédiction intra-image classiques.

Différentes perspectives sont envisageables dans la continuité de ce travail. La première concerne le regroupement de certains prédicteurs candidats issus du *template matching*. En effet, l'analyse de ces candidats a montré une forte similitude entre eux. Cette conformité constatée entre les candidats s'explique par la méthode de sélection par ressemblance du voisinage du *template matching*. Il nous semble alors judicieux de regrouper les candidats proches pour deux raisons. Tout d'abord, cette approche permettrait de réduire la taille du dictionnaire des indices destinés à identifier le prédicteur utilisé, pour ainsi réduire son coût de codage. Ensuite, le nombre de candidats regroupés au sein d'un même ensemble pourrait être un bon indicateur sur sa probabilité d'être choisi. Nous pouvons simplement présumer qu'il y a une relation proportionnelle entre eux : plus le nombre de candidats proches est important et plus la probabilité d'utiliser le prédicteur "moyen" de ces candidats est élevée. Cette démarche a déjà fait l'étude d'une expérimentation approfondie menée en annexe

des travaux de cette thèse. La méthode consiste à utiliser une méthode de *clustering* hiérarchique suivi d'un codage de Huffman pondéré par le nombre de candidats de chaque *cluster*. Le prédicteur d'un *cluster* est obtenu en moyennant tous ses candidats. Les résultats ont montré un faible gain en compression, cet algorithme de regroupement nécessiterait d'être perfectionné dans de futures investigations pour en améliorer l'efficacité.

Ensuite, il serait intéressant d'étudier le comportement de notre outil dans un contexte de codage inter-image. Dans ce cas, l'utilisation de notre méthode sort du cadre de la modélisation de texture pour relever de l'estimation de mouvement entre deux images d'une séquence vidéo. Les procédés d'estimation et de compensation de mouvement sont basés sur l'utilisation de vecteurs nécessitant le codage conjoint de deux informations (pour l'abscisse et l'ordonnée). Dans certains cas, notre approche pourrait se montrer productive puisqu'elle nécessite le codage d'un unique symbole appartenant à un ensemble de taille ajustable selon les besoins. Il s'agirait alors d'estimer le mouvement d'une région entre deux images par analyse de son voisinage connu.

Le deuxième axe de recherche est dédié à l'étude et à l'élaboration d'un nouveau procédé de compression intégré dans un schéma de codage d'image/vidéo classique. Dans un premier temps, nous avons élaboré un modèle théorique de régularisation d'image appliquée aux coefficients DCT associés. Nous l'avons ensuite décliné en deux d'applications concrètes de traitement d'image : la restauration de coefficients DCT perdus et l'amélioration d'une image décodée par reconstruction optimale des coefficients DCT quantifiés. Suite à ces premières expérimentations, nous avons mis au point un modèle de prédiction de coefficients DCT que nous avons intégré à un schéma de codage d'image de type JPEG. Enfin nous nous sommes intéressés à la réutilisation de ce modèle de prédiction dans un contexte de codage vidéo de type H.264/AVC, ce qui a constitué notre contribution majeure pour cette partie. Aux quatre étapes conventionnelles de la chaîne de traitement du procédé d'encodage, à savoir prédiction-transformation-quantification-codage, nous avons ajouté une méthode de prédiction d'un sous-ensemble des coefficients DCT résiduels de codage H.264/AVC à partir du sous-ensemble complémentaire, ce procédé se résume par l'appellation "prédiction DCT intra-bloc". Puisque le résidu de prédiction des coefficients DCT a, en moyenne, une énergie plus faible que les coefficients d'origine, nous constatons au final que notre méthode apporte un gain en compression non négligeable. De plus, notre nouvel outil de prédiction offre l'avantage de ne pas nécessiter le codage d'information de signalisation supplémentaire. En effet, nous l'avons voulu générique, c'est-à-dire qu'il s'applique indifféremment à tous les blocs.

Par cette démarche, nous avons choisi une approche différente de l'état de l'art portant sur les méthodes de régularisation d'image. En ce qui concerne les schémas typiques de codage d'image, la régularisation se caractérise soit par une méthode de post-traitement au décodage afin de lisser les imperfections liées à l'algorithme d'encodage, soit par un schéma de codage sans transformation par interpolation des pixels. Dans notre cas, nous avons choisi un compromis entre les deux approches : nous avons intégré un algorithme de régularisation d'image directement au sein d'une boucle de codage/décodage comme un outil de prédiction, et ce pour deux raisons. Premièrement, nous souhaitions conserver un schéma de codage très performant de type H.264/AVC basé sur une transformation DCT. Ensuite, nous voulions proposer une étape de codage faisant partie intégrante de la chaîne de traitement du processus d'encodage, avec comme objectif principal la réduction du débit.

La méthode de prédiction des coefficients de la DCT que nous avons développée a démontré sa capacité à améliorer la compression d'une image. Cependant, le procédé se révèle être d'une complexité calculatoire relativement importante. Comme la prédiction s'opère de façon synchrone au codage et

au décodage, cela rend plus complexe le procédé aux deux extrémités du canal de transmission. A la suite de ce travail, un axe de recherche à explorer concernerait la réduction du nombre d'opérations nécessaires pour effectuer cette nouvelle étape de prédiction. Enfin, notre étude a porté sur le codage purement intra-image. Il serait intéressant d'élaborer d'autres algorithmes basés sur notre travail et adaptés au contexte de codage inter-image.

De manière plus globale, l'objectif de cette thèse était de concevoir et d'expérimenter de nouvelles méthodes algorithmiques destinées à améliorer les performances du codeur vidéo H.264/AVC. De ce point de vue, notre contribution essentielle a été de développer deux nouvelles techniques performantes que nous avons ajoutées aux méthodes de codage existantes. Dans un avenir proche, le comité de normalisation JVT, poussé par des contraintes industrielles fortes, définira un nouveau standard de codage vidéo dont les spécifications sont essentiellement orientées vers l'augmentation du ratio de compression. Ce nouveau schéma de codage, en cours d'élaboration, reprend les principes fondamentaux qui ont fait le succès de H.264/AVC, entre autre la prédiction intra et inter-image et une transformation de type DCT. Pour accroître le rendement dans ce nouveau standard, l'idée majeure repose sur l'agrandissement de la taille des blocs de pixels (32×32 pixels, voire 64×64) pour deux raisons : l'augmentation croissante de la résolution des séquences vidéo augmente la corrélation entre les pixels, et la capacité de calcul des appareils actuels permet d'effectuer des traitements sur des blocs plus grands. La prise en compte de ce nouveau facteur modifie la perspective de notre travail. Pour le premier volet de notre travail basé sur un nouveau modèle de prédiction de texture, le traitement de blocs élargis va permettre de "capturer" des motifs de textures plus grands et plus complexes, ce qui au final contribuera à optimiser l'efficacité de notre méthode. Concernant la deuxième contribution de notre travail, à savoir la prédiction DCT intra-bloc, il serait intéressant d'étudier l'impact des blocs de grandes tailles sur l'efficacité du modèle de prédiction. Nos expérimentations ont montré que le sous-ensemble des coefficients du bloc à prédire dépend de la répartition de l'énergie au sein du bloc, et donc du mode de prédiction basé pixel utilisé pour décorréler l'information. Nous pouvons supposer qu'il en ira de même pour des blocs plus grands. En revanche, il serait intéressant d'étudier si la proportion de coefficients DCT à prédire peut augmenter en fonction de la taille du bloc, et dans quelle mesure. Dans ce cas, nous pourrions espérer accroître le gain en compression de notre solution.

Enfin, nous conclurons sur le travail effectue pendant cette thèse dans un contexte de normalisation. L'enjeu principal était de proposer aux organismes qui définissent les standards de nouvelles méthodes de codage vidéo plus performantes en terme de compression. Le travail que nous avons réalisé répond à cette attente et offre de nouvelles pistes de réflexion pour les schémas de codage futurs.

Bibliographie

[1] MPEG-4 : Information technology - Coding of audio-visual objects, ISO/IEC 14496-2. 2.2.1

[2] H.261 : Video codec for audiovisual services at p x 64 kbit/s, Recommendation H.261, ITU-T, 1993. 2.2.1

[3] H.263 : Video coding for low bitrate communication, First Recommendation H.263, ITU-T, 1995. 2.2.1

[4] MPEG-2 : Information Technology - Generic coding of moving pictures and associated audio information, ISO/IEC 13818-2, 1995. 2.2.1

[5] Recommendation ITU-R BT.601-5. Studio encoding paramters of digital television for standard 4 :3 and wide-screen 16 :9 aspect ratios, ITU-T, 1995. 2.3.1.1

[6] H.263+ : Video coding for low bitrate communication, Second recommendation H.263, ITU-T, 1998. 2.2.1

[7] H.263++ : Annex U, V, W and X, Extensions for recommendation H.263, ITU-T, 2000. 2.2.1

[8] Draft ITU-T Recommendation and final draft international standard of joint video specification, ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC, Joint Video Team of ITU-T and ISO/IEC JTC 1, 2003. 2.2.1

[9] Draft Text of H.264/AVC Fidelity Range Extensions Amendment, 2004. 2.3.2.1

[10] Proposed SMPTE standard for television : VC-1 compressed video bitstream format and decoding process, SMPTE draft standard for television, SMPTE technology committee C24 on video compression technology, 2005. 2.2.1

[11] Hyperconnectivity and the approaching zettabyte era. White paper, Cisco Visual Networking Index, June 2010. 1.2, 1.1, 1.2

[12] F. Alter, S. Durand et J. Froment : Adapted total variation for artifact free decompression of JPEG images. *J. Math. Imaging Vis.*, 23(2):199–211, 2005. 5.2.5, 5.2.5, 6.2, 6.3

[13] F. Alter, S.Y. Durand et J. Froment : Deblocking DCT-based compressed images with weighted total variations. *In IEEE International Conference on Acoustics, Speech, and Signal Processing, (ICASSP)*, volume 3, pages iii–221–4 vol.3, May 2004. 5.2.5

[14] L. Alvarez, P.-L. Lions et J.-M. Morel : Image selective smoothing and edge detection by nonlinear diffusion. ii. *SIAM Journal on Numerical Analysis*, 29(3):845–866, 1992. 5.1.2.3

[15] L. Ambrosio, N. Fusco et D. Pallara : *Functions of Bounded Variation and Free Discontinuity Problems*. Oxford Univ. Press, London, U.K., 2000. 5.1.2.2

[16] M. Ashikhmin : Synthesizing natural textures. *In Symposium on Interactive 3D Graphics*, pages 217–226, 2001. 3.3.2.1

[17] J.F Aujol et B. Matei : Structure and texture compression. Research Report 5076, INRIA, France, January 2004. 5.2.6

[18] J. Balle et M. Wien : Extended texture prediction for H.264/AVC intra coding. *In IEEE International Conference on Image Processing, ICIP*, volume 6, pages VI –93–VI –96, 16 2007-Oct. 19 2007. 3.5.2.6, 4.4

[19] Z. Bar-Joseph, R. El-Yaniv, D. Lischinski et M. Werman : Texture mixing and texture movie synthesis using statistical learning. *IEEE Transactions on Visualization and Computer Graphics, (TVCG)*, 7(2):120–135, 2001. 3.3.1

[20] M. Barlaud et C. Labit : *Compression et codage des images et vidéos*. Hermès Sciences Publications, 2002. 2.1

[21] J.R. Bergen et M.S. Landy : Computational modeling of visual texture segregation. *Computational Models of Visual Processing*, pages 253–271, 1991. 3.2

[22] M. Bertalmio, A.L. Bertozzi et G. Sapiro : Navier-stokes, fluid dynamics, and image and video inpainting. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR)*, 1:I–355–I–362 vol.1, 2001. 3.4.1

[23] M. Bertalmio, G. Sapiro, V. Caselles et C. Ballester : Image inpainting. *In ACM Siggraph, Computer Graphics Proceedings*, pages 417–424. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000. 3.4.1, 3.8, 5.2.4.1, 5.2.4.1

[24] M. Bertalmio, L. Vese, G. Sapiro et S. Osher : Simultaneous structure and texture image inpainting. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR)*, 2:II – 707–12 vol.2, jun. 2003. 5.2.4.1

[25] G. Bjontegaard : Calculation of average PSNR differences between RD curves. *In ITU-T SC16/Q6*, document VCEG-M33, USA, April 2001. 13th VCEG Meeting. 3.5.2.6, 4.3.2, 6.5.4

[26] G. Bjontegaard et K. Lillevold : Context-adaptive vlc coding of coefficients. Rapport technique, JVT document JVT-C028, 2002. 2.3.2.4

[27] L. Blanc-Feraud, P. Charbonnier, G. Aubert et M. Barlaud : Nonlinear image processing : modeling and fast algorithm for regularization with edge detection. *IEEE International Conference on Image Processing, (ICIP)*, 1:474, 1995. 5.1.2.3

[28] P. Blomgren et T.F. Chan : Color TV : total variation methods for restoration of vector-valued images. *International Conference on Image Processing, (ICIP)*, 7(3):304 –309, mar. 1998. 5.1.2.3

[29] J.S. De Bonet : Multiresolution sampling procedure for analysis and synthesis of texture images. *Computer Graphics*, 31(Annual Conference Series):361–368, 1997. 3.3, 3.3.1, 3.3

[30] J.S. De Bonet et P.A. Viola : A non-parametric multi-scale statistical model for natural images. *In* Michael I. Jordan, Michael J. Kearns et Sara A. Solla, éditeurs : *Advances in Neural Information Processing Systems*, volume 10, pages 773–779. MIT Press, 1998. 3.2, 3.2

[31] R. Bornard, E. Lecan, L. Laborelli et J.-H. Chenot : Missing data correction in still images and image sequences. *In Proceedings of the tenth ACM international conference on Multimedia*, MULTIMEDIA '02, pages 355–361, New York, NY, USA, 2002. ACM. 3.4.1

[32] P.J. Burt : Fast filter transform for image processing. *Computer Graphics and Image Processing*, 16(1):20 – 51, 1981. 3.3.1

[33] F. J. Canny : A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence, (TPAMI)*, 8(6):679–698, 1986. 5.2.6

[34] A. Chambolle : An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20(1):89–97, January 2004. 5.1.3.1

[35] A. Chambolle et P. L. Lions : Image recovery via total variation minimization and related problems. *Numerische Mathematik*, 76(2):167–188, 1997. 5.1.2.2

[36] T.F. Chan et S. Esedoglu : Aspects of total variation regularized l^1 function approximation. *SIAM Journal on Applied Mathematics*, 65:1817–1837, 2005. 5.2.3

[37] T.F. Chan, G.H. Golub et P. Mulet : A nonlinear primal-dual method for total variation-based image restoration. *SIAM Journal on Scientific Computing, (SISC)*, 20(6):1964–1977, 1999. 5.1.3.1, 5.3

[38] T.F. Chan, S.H. Kang et J.H. Shen : Euler's elastica and curvature based inpaintings. *SIAM Journal of Applied Mathematics*, 63:564–592, 2002. 3.4.1, 5.2.4.1, 5.2.4.2

[39] T.F. Chan et J. Shen : Non-texture inpaintings by curvature-driven diffusions. *Journal of Visual Communication and Image Representation*, 12(4):436–449, 2001. 3.4.1, 5.1.2.3, 5.2, 5.2.4.1

[40] T.F. Chan et J. Shen : Mathematical models for local non-texture inpaintings. *SIAM Journal of Applied Mathematics*, 63(3):1019–1043, 2002. 5.2.4.1, 5.2.4.1, 5.2.6

[41] T.F. Chan, J. Shen et H.M. Zhou : Total variation wavelet inpainting. *Journal of Mathematical Imaging and Vision*, 25:107–125, 2006. 5.2.4.2, 5.2.4.2, 5.2.4.2

[42] T.F. Chan et C.K. Wong : Total variation blind deconvolution. *IEEE International Conference on Image Processing, (ICIP*, 7(3):370–375, March 1998. 5.2.2, 5.2.2

[43] T.F. Chan et H.M. Zhou : Feature preserving lossy image compression using nonlinear PDE's. *In SPIE Proceedings on Advanced Signal Processing Algorithms, Architectures, and Implementations VIII*, pages 316–327, 1998. 5.2.5

[44] T.F. Chan et H.M. Zhou : Optimal constructions of wavelet coefficients using total variation regularization in image compression. *In Image Compression, CAM Report, No. 00-27*, 2000. 5.2.4.2, 5.2.5

[45] T.F. Chan et H.M. Zhou : Total variation improved wavelet thresholding in image compression. *In IEEE International Conference on Image Processing, (ICIP)*, pages 391–394, 2000. 5.2.5

[46] P. Charbonnier, L. Blanc-Feraud, G. Aubert et M. Barlaud : Two deterministic half-quadratic regularization algorithms for computed imaging. *IEEE International Conference on Image Processing, (ICIP)*, 2:168 –172, nov. 1994. 5.1.2.3

[47] Y. Chen et T. Wunderli : Adaptive total variation for image restoration in bv space. *Journal of Mathematical Analysis and Applications*, 272:117–137, 2002. 5.2.5

[48] A. Cohen et B. Matei : Compact representation of images by edge adapted multiscale transforms. *IEEE International Conference on Image Processing, (ICIP)*, 1:8 –11, 2001. 5.2.6

[49] P.L. Combettes et J.-C. Pesquet : Image restoration subject to a total variation constraint. *IEEE International Conference on Image Processing, (ICIP)*, 13(9):1213–1222, sep. 2004. 5.1.2.2

[50] A. Criminisi, P. Perez et K. Toyama : Region filling and object removal by exemplar-based image inpainting. *IEEE International Conference on Image Processing, (ICIP)*, 13(9):1200–1212, Sept. 2004. (document), 3.4.1, 3.1, 3.8, 3.4.2, 3.4.3, 3.9, 3.10, 3.5.2.2, IV

[51] J. Darbon et M. Sigelle : A fast and exact algorithm for total variation minimization. *In Pattern Recognition and Image Analysis*, volume 3522 de *Lecture Notes in Computer Science*, pages 717–765. Springer Berlin / Heidelberg, 2005. 5.1.3.1

[52] R. Deriche et O. Faugeras : Les EDP en traitement des images et vision par ordinateur. Rapport de Recherche 2697, INRIA Sophia Antipolis, 1995. 5.1.2.3

[53] R. Distasi, M. Nappi et S. Vitulano. : Image compression by B-tree triangular coding. *IEEE Transactions on Communications*, 45(9):1095–1100, 1997. 5.2.6

[54] I. Drori, D. Cohen-Or et H. Yeshurun : Fragment-based image completion. *ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH*, 22(3):303–312, 2003. 3.4.1, 3.4.1

[55] A. Dumitras et B.G. Haskell : An encoder-decoder texture replacement method with application to content-based movie coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(6):825–840, 2004. 3.5.1

[56] A.A. Efros et W.T. Freeman : Image quilting for texture synthesis and transfer. *In Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 341–346. ACM, 2001. 3.3.2.2, 3.6

[57] A.A. Efros et T.K. Leung : Texture synthesis by non-parametric sampling. *IEEE International Conference on Computer Vision, (ICCV)*, 2:1033–1038, 1999. 3.3.2.1, 3.4, 3.3.2.1, 3.5, 3.4.1, 5.2.4.1

[58] M. Elad, J.-L Starck, D. Donoho et P. Querre : Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA). *Journal on Applied and Computational Harmonic Analysis, (ACHA)*, 19:340–358, November 2005. 5.2.4.1

[59] X. Fan, Y. Lu et W. Gao : A novel coefficient scanning scheme for directional spatial prediction-based image compression. *IEEE International Conference on Multimedia and Expo, (ICME)*, 2:557–560, 2003. 6.5.3

[60] W. Freeman : *Steerable filters and the local analysis of image structure*. Thèse de doctorat, Massachusetts Institute of Technology, 1992. 3.2

[61] W.T. Freeman et E.H. Adelson : The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence, (TPAMI)*, 13(9):891–906, 1991. 3.3.1

[62] I. Galić, J. Weickert, M. Welk, A. Bruhn, A. Belyaev et H.-P. Seidel : Towards PDE-based image compression. Lecture notes in computer science, 2005. 5.2.6

[63] I. Galić, J. Weickert, M. Welk, A. Bruhn, A. Belyaev et H.-P. Seidel : Image compression with anisotropic diffusion. *Journal of Mathematical Imaging and Vision*, 31:255–269, 2008. 5.2.6

[64] D. Geman et G. Reynolds : Constrained restoration and the recovery of discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence, (TPAMI)*, 14(3):367–383, 1992. 5.1.2.3

[65] G. Gilboa, N. Sochen et Y.Y. Zeevi : Texture preserving variational denoising using an adaptive fidelity term. Proceedings of the IEEE Workshop on Variational and Level Set Methods (VLSM), Oct. 2003. 5.1.2.3, 5.2.1

[66] T. Goldstein et S. Osher : The split bregman method for l1-regularized problems. *SIAM Journal on Imaging Sciences*, 2(2):323–343, 2009. 5.1.3.1, 5.3

[67] S.W. Golomb : Run-length encoding. *IEEE Transactions on Information Theory*, 12:399–401, 1966. 2.3.1.6

[68] S.J. Gortler, R. Grzeszczuk, R. Szeliski et M.F. Cohen : The lumigraph. *In SIGGRAPH '96 : Proceedings of the 23rd annual conference on Computer Graphics and Interactive Techniques*, pages 43–54, New York, NY, USA, 1996. ACM. 1

[69] F. Guichard et F. Malgouyres : Total variation based interpolation. *In Proceedings of European signal processing conference, (Eusipco)*, pages 1741–1744, 1998. 5.2.4.1, 5.2.4.1

[70] Y. Guo, Y.-K. Wang et H. Lis : Priority-based template matching intra prediction. *In IEEE International Conference on Multimedia and Expo, (ICME)*, pages 1117–1120, 2008. 3.5.2.2, 4.3.1.1

[71] P. Harrison : A non-hierarchical procedure for re-synthesis of complex textures. *In WSCG Conference proceedings*, pages 190–197, 2001. 3.4.1

[72] L. He, A. Marquina et S. Osher : Blind deconvolution using tv regularization and bregman iteration. *International Journal of Imaging Systems and Technology*, 15:74–83, 2005. 5.2.2, 5.3

[73] D.J. Heeger et J.R. Bergen : Pyramid-based texture analysis/synthesis. *In IEEE International Conference on Image Processing, (ICIP)*, volume 3, 1995. 3.3.1, 3.2

[74] D.A. Huffman : A method for the construction of minimum redundancy codes. *Proceedings of the Institute of Radio Engineers*, 40:1098–1101, 1952. 2.3.1.6

[75] A.K. Jain : *Fundamentals of digital image processing*. Prentice-Hall, 1989. 3.2

[76] A. Jakulin : Baseline JPEG and JPEG2000 artifacts illustrated. http ://www.stat.columbia.edu/ jakulin/jpeg/artifacts.htm, 2004. 5.2.5

[77] J. Jia, E.-K. Jung et H.-K. Kim : Adaptive transform coefficient scan for H.264 intra coding. *IEICE Transactions on Information and Systems*, 90(10):1709–1711, 2007. 6.5.3

[78] J. Jia et C.-K. Tang : Image repairing : robust image synthesis by adaptive ND tensor voting. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR)*, 1:643–650, 2003. 3.4.1

[79] A.K. Katsaggelos et K.T. Lay : Simultaneous blur identification and image restoration using the em algorithm. *SPIE Conference on Visual Communications and Image Processing*, 3:1474–1485, 1989. 5.2.2

[80] S. Kondo, H. Sasai et S. Kadono : Tree structured hybrid intra prediction. *IEEE International Conference on Image Processing, (ICIP)*, 1:473–476, 2004. 4.4

[81] D. Kundur et D. Hatzinakos : A novel blind deconvolution scheme for image restoration using recursive filtering. *IEEE Transactions on Signal Processing*, 46(2):375–390, feb. 1998. 5.2.2

[82] V. Kwatra, A. Schödl, I. Essa, G. Turk et A. Bobick : Graphcut textures : image and video synthesis using graph cuts. *ACM Transactions on Graphics, (TOG)*, 22(3):277–286, 2003. 3.3.2.2, 3.5.1

[83] R. L. Lagendijk et J. Biemond : *Iterative Identification and Restoration of Images*. Springer-Verlag, 2001. 5.2.2

[84] Y.-L. Lee, K.-H. Han, D.-G. Sim et J. Seo : Adaptive scanning for H.264/AVC intra coding. *ETRI Journal*, 28(5):668–671, 2006. 6.5.3

[85] L. Liang, C. Liu, Y. Xu, B. Guo et H.-Y. Shum : Real-time texture synthesis by patch-based sampling. *ACM Transactions on Graphics (TOG)*, 20:127–150, 2001. 3.3.2.2

[86] A. Likas et N.P. Galatsanos : A variational method for bayesian blind image deconvolution. *IEEE International Conference on Image Processing, (ICIP)*, 2:973–976, 2003. 5.2.2

[87] W.-C. Lin, J.H. Hays, C. Wu, V. Kwatra et Y. Liu : A comparison study of four texture synthesis algorithms on regular and near-regular textures. *In ACM SIGGRAPH*, 2004. 3.1, 3.3.3

[88] P. List, A. Joch, J. Lainema, G. Bjontegaard et M. Karczewicz : Adaptive deblocking filter. *IEEE Transaction on Circuits and Systems for Video Technology*, 13:614–619, 2003. 2.3.1.7

[89] D. Liu, X. Sun et F. Wu : Edge-based inpainting and texture synthesis for image compression. *In IEEE International Conference on Multimedia and Expo, (ICME)*, pages 1443–1446, 2007. 3.5.1

[90] D. Liu, S. Xiaoyan, F. Wu, S. Li et Y.-Q. Zhang : Image compression with edge-based inpainting. *IEEE transaction on Circuits and Systems for Video Technology*, 17(5):639–644, 2007. 3.5.1

[91] L. Luo, F. Wu, S. Li, Z. Xiong et Z. Zhuang : Advanced motion threading for 3d wavelet video coding. *In Signal Processing : Image Communication*, volume 19, pages 601–616, 2004. 3.5.1

[92] S. Mallat : *A Wavelet Tour of Signal Processing*. Academic Press, 2 édition, 1999. 5.2.6

[93] H. S. Malvar, A. Hallapuro, M. Karczewicz et L. Kerofsky : Low-complexity transform and quantization in h.264/avc. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):598–603, 2003. 2.3.2.3

[94] D. Marpe, H. Schwarz, G. Blattermann, G. Heising et T. Wiegb : Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):620–636, 2003. 2.3.2.4, 2.3.2.4

[95] D. Marpe, T. Wiegand et S. Gordon : H.264/MPEG4-AVC fidelity range extensions : Tools, profiles, performance, and application areas. *International Conference on Image Processing*, 1:593–596, 2005. 2.3.2.1

[96] S. Masnou : Disocclusion : a variational approach using level lines. *IEEE Transactions on Image Processing*, 11(2):68–76, 2002. 5.2.4.1

[97] S. Masnou et J.-M. Morel : Level lines based disocclusion. *IEEE International Conference on Image Processing, (ICIP)*, 3:259–263, 1998. 5.2.4.1

[98] Y. Meyer : *Oscillating Patterns in Image Processing and Nonlinear Evolution Equations : The Fifteenth Dean Jacqueline B. Lewis Memorial Lectures*. American Mathematical Society, 2001. 5.2.3, 5.2.3

[99] M. Moinard, I. Amonou, P. Duhamel et P. Brault : A set of template matching predictors for intra video coding. *In IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pages 1422 –1425, 2010. 4.4

[100] P. Ndjiki-Nya, T. Hinz, C. Stuber et T. Wiegand : A content-based video coding approach for rigid and non-rigid textures. *In IEEE International Conference on Image Processing, (ICIP)*, pages 3169–3172, 2006. 3.5.1

[101] P. Ndjiki-Nya, T. Hinz et T. Wiegand : Generic and robust video coding with texture analysis and synthesis. *In IEEE International Conference on Multimedia and Expo, (ICME)*, pages 1447–1450, 2007. 3.5.1

[102] P. Ndjiki-Nya, M. Kootz et T. Wiegand : Automatic detection of video synthesis related artifacts. *IEEE International Conference on Acoustics, Speech, and Signal Processing, (ICASSP)*, 3:733–736, 2004. 3.5.1

[103] P. Ndjiki-Nya, C. Stuber et T. Wiegand : A new generic texture synthesis approach for enhanced H.264/MPEG4-AVC video coding. *In 9th international workshop, VLBV*, volume 3893, pages 121–128, 2005. 3.5.1

[104] P. Ndjiki-Nya, C. Stuber et T. Wiegand : Texture synthesis method for generic video sequences. *IEEE International Conference on Image Processing, (ICIP)*, 3:397–400, 2007. 3.3.2.2, 3.5.1

[105] M.K. Ng, R.J. Plemmons et S. Qiao : Regularization of RIF blind image deconvolution. *IEEE Transactions on Image Processing, (ICIP)*, 9(6):1130 –1134, 2000. 5.2.2

[106] J.-R. Ohm : *Multimedia Communication Technology*. Springer, 2004. 3.5.1

[107] T. Ojala, M. Pietikainen et D. Harwood : A comparative study of texture measures with classification based on feature distributions. *Pattern Recognition*, 29(1):51–59, 1996. 3.2

[108] M. Oliveira, B. Bowen, R. McKenna et Y.-S. Chang : Fast digital image inpainting. *In* M. H. Hamza, éditeur : *Proceedings of the International Conference on Visualization, Imaging and Image Processing, (VIIP)*, pages 261–266. ACTA Press, 2001. 5.2.4.1

[109] E. Ong, W. Lin, Z. Lu et S. Yao : Colour perceptual video quality metric. *IEEE International Conference on Image Processing, (ICIP)*, 3:1172–1177, 2005. 3.5.1

[110] F. Ono, W. Rucklidge, R. Arps et C. Constantinescu : JBIG2-the ultimate bi-level image coding standard. *IEEE International Conference on Image Processing, (ICIP)*, 1:140 –143, 2000. 5.2.6

[111] P. Perona et J. Malik : Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990. 5.1.2.3

[112] M. Pietikäinen : Image analysis with local binary patterns. *In Image Analysis*, volume 3540 de *Lecture Notes in Computer Science*, pages 115–118. Springer Berlin / Heidelberg, 2005. 3.2

[113] K. Popat et R.W. Picard : Novel cluster-based probability model for texture synthesis, classification, and compression. *In Visual Communications and Image Processing, (VCIP)*, pages 756–768, 1993. 3.3.2.1

[114] M. Porat et Y.Y. Zeevi : Localized texture processing in vision : analysis and synthesis in the gaborian space. *IEEE Transactions on Biomedical Engineering*, 36(1):115 –129, 1989. 3.3.1

[115] J. Portilla et E. Simoncelli : A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision*, 40(1):49–70, 2000. 3.2, 3.3.1

[116] S.D. Rane, G. Sapiro et M. Bertalmio : Structure and texture filling-in of missing image blocks in wireless transmission and compression applications. *IEEE Transactions on Image Processing, (ICIP)*, 12(3):296–303, 2003. 5.2.4.1

[117] K. R. Rao et P. Yip : *Discrete cosine transform : algorithms, advantages, applications*. Academic Press Professional, 1990. 2.3.1.3

[118] I.E. Richardson : *H.264 and MPEG-4 Video Compression : Video Coding for Next Generation Multimedia*. Wiley, 1 édition, August 2003. 2.3.1.1, 2.3.2.1, 2.3.2.3, 6.1.2, 6.5.1, 6.5.3, 6.5.4

[119] J. Rissanen : Generalized kraft inequality and arithmetic coding. *IBM Journal of Research and Development*, 20(3):198–203, 1976. 2.3.1.6

[120] L. Rudin, P.-L. Lions et S. Osher : Multiplicative denoising and deblurring : Theory and algorithms. *In Geometric Level Set Methods in Imaging, Vision, and Graphics*, pages 103–119. Springer New York, 2003. 5.1.1

[121] L. I. Rudin, S. Osher et E. Fatemi : Nonlinear total variation based noise removal algorithms. *Physica D : Nonlinear Phenomena*, 60(1-4):259–268, 1992. 5.1.2.2, 5.1, 5.1.3.1, 5.1.3.1, 5.2.1

[122] L.I. Rudin : Images, numerical analysis of singularities and shock filters. Rapport technique, Technical Report. California Institute of Technology, 1987. 5.1.2.2

[123] P. Salembier et F. Marques : Region-based representations of image and video : segmentation tools for multimedia services. *IEEE Transactions on Circuits and Systems for Video Technology*, 9:1147–1169, 1999. 2.3.3.1

[124] G. Sapiro et D. Ringach : Anisotropic diffusion of multivalued images. *In ICAOS*, volume 219, pages 134–140. Springer Berlin / Heidelberg, 1996. 5.1.2.3

[125] H. Schwarz, T. Hinz et K. Suehring : Joint scalable video model 9.7 (JSVM 9.7). Rapport technique, Joint Video Team, 2005. 4.3.2, 6.5.4

[126] H. Schwarz, D. Marpe et T. Wiegand : Overview of the scalable extension of the H.264 / MPEG-4 AVC Video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 2007. 6.6

[127] C.E. Shannon : A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423,623–656, 1948. 2.3.1.6

[128] T. Sikora et B. Makai : Shape-adaptive DCT for generic coding of video. *IEEE Transaction on Circuits and Systems for Video Technology*, 5(1):59–62, 1995. 2.3.3.1

[129] E.P. Simoncelli, W.T. Freeman, E.H. Adelson et D.J. Heeger : Shiftable multiscale transforms. *IEEE Transactions on Information Theory*, 38(2):587–607, 1992. 3.3.1

[130] J. R. Smith : *Integrated Spatial and Feature Image Systems : Retrieval, Analysis and Compression.* Thèse de doctorat, Columbia University, 1997. 3.2

[131] F. Sroubek et J. Flusser : Multichannel blind iterative image restoration. *IEEE Transactions on Image Processing, (ICIP)*, 12(9):1094–1106, 2003. 5.2.2

[132] K. Sugimoto, M. Kobayashi, Y. Suzuki, S. Kato et Choong S.B. : Inter frame coding with template matching spatio-temporal prediction. *IEEE International Conference on Image Processing, (ICIP)*, 1:465–468, 2004. 3.5.2.5

[133] G.J. Sullivan, P. Topiwala et A. Luthra : The H.264/AVC advanced video coding standard : Overview and introduction to the fidelity range extensions. *In SPIE conference on Applications of Digital Image Processing XXVII*, pages 454–474, 2004. 2.2.1, 2.3.2.1

[134] G.J. Sullivan et T. Wiegand : Rate-distortion optimization for video compression. *Signal Processing Magazine, IEEE*, 15(6):74–90, 1998. 3.5.2

[135] Y. Suzuki, C.S. Boon et S. Kato : Block-based reduced resolution inter frame coding with template matching prediction. *In IEEE International Conference on Image Processing, (ICIP)*, pages 1701–1704, 2006. 3.5.2.5

[136] Y. Suzuki, C.S. Boon et T.K. Tan : Video encoding scheme employing intra and inter prediction based on averaged template matching predictors. *IEICE Transactions on Information and Systems*, 91(4):1127–1134, 2008. 3.5.2.5

[137] T.K. Tan, C.S. Boon et Y. Suzuki : Intra prediction by template matching. *In IEEE International Conference on Image Processing, (ICIP)*, pages 1693–1696, 2006. 3.5.2.1, 3.12, 3.5.2.1, 3.5.2.2, 3.5.2.3, 3.5.2.4, 4.9

[138] T.K. Tan, C.S. Boon et Y.Suzuki : Intra prediction by averaged template matching predictors. *In IEEE Consumer Communications and Networking Conference, (CCNC)*, pages 405–409, 2007. 3.5.2.3, 4.1.2, 4.9

[139] D. S. Taubman et M. W. Marcellin : *JPEG2000 : image compression fundamentals, standards, and practice*. Kluwer Academic Publishers, 2002. 5.2.5

[140] J. Teuhola : A compression method for clustered bit-vectors. *Information Processing Letters*, 7:308–311, 1978. 2.3.1.6

[141] A.N. Tikhonov : Regularization oh incorrectly posed porblems. *Soviet Mathematics Doklady*, 4:1624–1627, 1963. 5.1.2.1, 5.1.2.1, 5.1

[142] D. Tschumperlé et R. Deriche : Vector-valued image regularization with PDEs : A common framework for different applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):506–517, 2005. 5.1.2.3, 5.2.4.1, 5.2.5, 5.3

[143] David Tschumperlé : Curvature-preserving regularization of multi-valued images using PDE's. *In European Conference on Computer Vision, (ECCV)*, pages 295–307, 2006. 3.4.1, 5.2.4.1, 5.2.4.1, 5.3

[144] David Tschumperlé : Fast anisotropic smoothing of multi-valued images using curvature-preserving PDE's. *International Journal of Computer Vision*, 68:65–82, 2006. 5.2.4.1, 5.3

[145] L.A. Vese et S.J. Osher : Modeling textures with total variation minimization and oscillating patterns in image processing. *Journal of Scientific Computing*, 19:553–572, 2002. 5.2.3, 5.2.3, 5.2.3, 5.2.4.1, 5.4

[146] L.A. Vese et S.J. Osher : Color texture modeling and color image decomposition in a variational-PDE approach. *In Symposium on Symbolic and Numeric Algorithms for Scientific Computing, (SYNASC)*, pages 103–110. IEEE Computer Society, 2006. 5.2.3

[147] M. Vetterli : Fast 2-D discrete cosine transform. *In IEEE International Conference on Acoustics, Speech, and Signal Processing, (ICASSP)*, pages 1538–1541, 1985. 2.3.1.3

[148] J.S. Vitter : Design and analysis of dynamic huffman codes. *Journal of the ACM*, 34:825–845, 1987. 2.3.1.6

[149] C. R. Vogel et M. E. Oman : Iterative methods for total variation denoising. *SIAM Journal on Scientific Computing (SISC)*, 17(1):227–238, 1996. 5.1.3.1, 5.1.3.1, 5.3

[150] G. K. Wallace : The JPEG still picture compression standards. *Communication of ACM*, 34(4):30–44, 1991. 5.2.5

[151] C. Wang, X. Sun, F. Wu et H. Xiong : Image compression with structure-aware inpainting. *In IEEE International Symposium on Circuits and Systems, (ISCAS)*, 2006. 3.5.1

[152] L.-Y. Wei et M. Levoy : Fast texture synthesis using tree-structured vector quantization. *In Proceedings of the Conference on Computer Graphics and Interactive Techniques, SIGGRAPH*, pages 479–488, 2000. 3.3.2.1, 5.2.4.1

[153] T.A. Welch : A technique for high-performance data compression. *IEEE Computer*, 17:8–19, 1984. 2.3.1.6

[154] I.H. Witten, R.M. Neal et J.G. Cleary : Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987. 2.3.1.6

[155] R.C. Wrede et M. Spiegel : *Advanced Calculus*. Schaum's Outline, 2002. 5.1.2.3

[156] X. Wu, Q. Sun, K. Zhang et L. Yu : Modeling natural image for estimating DCT coefficient properties of intra prediction. *In IEEE International Conference on Multimedia and Expo, (ICME)*, pages 476–479, 2007. 6.5.3

[157] H. Yamauchi, J. Haber et H.-P. Seidel : Image restoration using multiresolution texture synthesis and image inpainting. *In Proceedings of Computer Graphics International*, pages 120–125, 2003. 5.2.4.1

[158] J. Yang, B. Yin et N. Zhang : A block-matching based intra frame prediction for H.264/AVC. *In IEEE International Conference on Multimedia and Expo, (ICME)*, pages 705–708, July 2006. 3.5.2.1

[159] W. Yin, D. Goldfarb et S. Osher : Image cartoon-texture decomposition and feature selection using the total variation regularized l 1 functional. *In Variational, Geometric, and Level Set Methods in Computer Vision*, pages 73–84. Springer, 2005. 5.2.3

[160] W. Yin, D. Goldfarb et S. Osher : A comparison of three total variation based texture extraction models. *Journal of Visual Communication and Image Representation*, 18(3):240–252, 2007. 5.2.3

[161] Y.-L. You et M. Kaveh : Blind image restoration by anisotropic regularization. *IEEE Transactions on Image Processing*, 8(3):396–407, mar. 1999. 5.2.2

[162] S. L. Yu et C. Chrysafis : New intra prediction using intra-macroblock motion compensation. *In JVT-C151, Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG Meeting*, May 2002. 3.5.2.4, 3.5.2.6, 4.4

[163] Y. Zheng, P. Yin, E.O. Divorra, X. Li et C. Gomila : Intra prediction using template matching with adaptive illumination compensation. *In IEEE International Conference on Image Processing, (ICIP)*, pages 125–128, 2004. 3.5.2.4

[164] S. Zhong : Image compression by optimal reconstruction. United States Patent 5,534,925, July 1996. Cognitech Inc. 5.2.5

[165] H. Zhou et J. Zheng : Adaptative patch size determination for patch-based image completion. *In IEEE International Conference on Image Processing, (ICIP)*, pages 421–424, 2010. 3.4.1

[166] C. Zhu, X. Sun, F. Wu et H. Li : Video coding with spatio-temporal texture synthesis. *In IEEE International Conference on Multimedia and Expo, (ICME)*, pages 112–115, 2007. 3.5.1

[167] C. Zhu, X. Sun, F. Wu et H. Lis : Video coding with spatio-temporal texture synthesis and edge-based inpainting. *IEEE International Conference on Multimedia and Expo, (ICME)*, 1:813–816, 2008. 3.5.1

[168] J. Ziv et A. Lempel : Compression of individual sequences via variable rate encoding. *Communication of the ACM*, 24:520–536, 1978. 2.3.1.6

Annexe

Illustration de deux méthodes de restauration basées patch

Image d'origine.

Masque (en rouge) de la région à restaurer.

Méthode de Criminisi [50]. Taille des patchs : 11×11 pixels, fenêtre de recherche : 300×300 pixels.

Contribution apportée (*cf.* Chapitre 3). Taille des patchs : 11×11 pixels, fenêtre de recherche : 300×300 pixels.

Méthode de Criminisi.

Contribution apportée.

Agrandissement des résultats.

Image d'origine

Masque (en rouge) de la région à restaurer.

Méthode de Criminisi.

Contribution apportée.

Image d'origine (recadrée).

Masque (en rouge) de la région à restaurer.

Méthode de Criminisi.

Contribution apportée.

Codage vidéo hybride basé contenu par analyse/synthèse de données

RESUME : Les travaux de cette thèse sont destinés à la conception d'outils algorithmiques permettant d'accroître le facteur de compression des standards actuels de codage vidéo, tels que H.264/AVC. Pour cela, une étude préalable portant sur un ensemble de méthodes de restauration d'image a permis d'identifier et d'inspecter deux axes de recherche distincts.

La première partie est fondée sur des méthodes d'analyse et de synthèse de texture. Ce type de procédé, aussi connu sous le nom de *template matching*, est couramment utilisé dans un contexte de codage vidéo pour prédire une portion de la texture de l'image suite à l'analyse de son voisinage. Nous avons cherché à améliorer le modèle de prédiction en prenant en compte les spécificités d'un codeur vidéo de type H.264/AVC. En particulier, la fonction débit/distorsion utilisée dans les schémas de codage vidéo normatifs se base sur une mesure objective de la qualité. Ce mécanisme est par nature incompatible avec le concept de synthèse de texture, dont l'efficacité est habituellement mesurée selon des critères purement perceptuels. Cette contradiction a motivé le travail de notre première contribution.

La deuxième partie des travaux de cette thèse s'inspire des méthodes de régularisation d'image basée sur la minimisation de la variation totale. Des méthodes ont été élaborées originellement dans le but d'améliorer la qualité d'une image en fonction de la connaissance *a priori* des dégradations qu'elle a subies. Nous nous sommes basés sur ces travaux pour concevoir un modèle de prédiction des coefficients transformés obtenus à partir d'une image naturelle, qui a été intégré dans un schéma de codage vidéo conventionnel.

Mots clés : codage vidéo, MPEG-4, H.264/AVC, prédiction intra-image, synthèse d'image, *template matching*, régularisation d'image, prédiction de coefficients transformés.

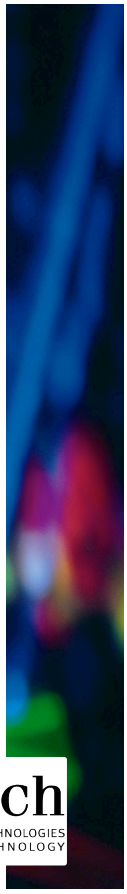
Content based hybrid video coding with data analysis/synthesis

ABSTRACT : This thesis is about the design of new algorithm tools that improve the compression ratio of current video coding standards, such as H.264/AVC. To reach this goal, a preliminary study on a set of image restoration methods identified two distinct lines of research.

The first is based on methods of texture analysis and synthesis. This kind of method, also known as template matching, is commonly used in video coding contexts to predict a portion of an image texture from an analysis of its neighborhood. We tried to improve the prediction model by taking into account the specificities of video encoders such as H.264/AVC. In particular, the rate-distortion function used in video coding standards is commonly based on an objective measure. This mechanism is inherently incompatible with the concept of texture synthesis, whose effectiveness is usually measured perceptually. It was this contradiction that motivated this first line of research.

The second is inspired by image regularization methods based on total variation minimization. These methods were originally developed in order to improve the quality of an image according to prior knowledge of its damage. Starting from this work, we designed a predictive model of transformed coefficients obtained from a natural image, which is integrated into a conventional video encoding scheme.

Keywords : video coding, MPEG-4, H.264/AVC, intra-frame prediction, image synthesis, template matching, image regularization, transformed coefficient prediction.



TELECOM ParisTech

EDITE

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY